Sequential Nonlinear-Programming Approach to Thermal-Aware VLSI Floorplanning using Multi-boundary Shapes

Paul B. Morton (pmorton@monarchip.com)

Abstract

In this paper we develop and implement sequential nonlinear-programming methods for solving the thermal-aware soft-macro VLSI floorplanning problem with IO-block placement and a dynamic floorplan-boundary. We develop a multi-stage nonlinear-programming approach to this floorplanning problem. We break the floorplanning process into two main stages, a simplified first-stage, which omits any consideration of the floorplan boundary and IO-block placement avoiding introducing adverse bias due to premature consideration of these elements, and a detailed second-stage, which uses information extracted from the first-stage to minimize or eliminate adverse bias when incorporating the floorplan boundary and IO-blocks back into the floorplanning process. Further, we develop methods, universal to all axis-aligned rectangle packing problems, for providing and maintaining macro-block mobility throughout the optimization process allowing our analytical floorplanner to consistently generate high quality floorplans. Finally, we develop a method for making floorplanning thermally aware by modeling each macro-block's thermal-signature as an additional, curvilinear, macro-block boundary from which we can derive and enforce an additional set of nonoverlap constraints between those boundaries.

Key words: VLSI floorplanning, thermal-aware, soft-macro, non-convex, sequential nonlinear-programming, Analytical Computational-Geometry, ACGL.

1 Introduction

In this paper we will develop methods for solving VLSI floorplanning problems. Specifically, we will be developing methods for thermal-aware soft-macro floorplanning with IO-block placement and a dynamic floorplan-boundary. That is, we want to determine the location and aspect ratio of each macro-block, the shape and size of the floorplan boundary, and the location on the floorplan boundary for each IO-block. This approach to floorplanning is suitable for full-custom design for high-performance and/or high-volume chips where the floorplan boundary and the placement of IO-blocks are determined as part of the process in order to obtain the best possible timing, power, and/or chip-area results.

As the main design variables in floorplanning are continuous it naturally leads to optimization problems which are inherently continuous in nature. Consequently, we will be developing analytical optimization methods to solve this floorplanning problem. To simplify our presentation and focus on exploring the floorplanning problem's optimizer-independent fundamental-behaviors we will develop our methods such that they can be implemented using "off the shelf" commercial or open source nonlinear-optimization software [1, 2, 3, 4] avoiding the need to implement a custom nonlinear optimizer. In particular, we will implement our methods using IPOPT [5, 1]. Note, however, that our methods have been designed to be easily reformulated, using an exterior quadratic-penalty function or an interior barrier-function, to effectively work with customized Nesterov accelerated-gradient optimization methods [6] used in many analytical placement tools [7, 8, 9].

Because the floorplanning problem is NP-hard it cannot be formulated as a convex optimization problem. In particular, the various nonoverlap and abutment constraints associated with the macro and IO blocks are chiefly responsible for the computationally intractable nature of the floorplanning problem and its formulation as a nonconvex optimization problem. Because of its nonconvex nature, solution quality is very sensitive to the formulation details and the initial optimization-variable values provided to the optimizer. This, in turn, means that finding good solutions to this floorplanning problem requires a deeper understanding of the nature of the problem and how it interacts with the optimizer than would be necessary for a problem that could be formulated as a convex optimization. Understanding and exploiting the nature of the problem and its interaction with the optimizer is central to developing high-quality solutions and, as we will demonstrate, generally leads to methods which solve a sequence of related nonlinear optimization problems. For the floorplanning problem, as we will demonstrate, the keys to developing these methods are a flexible and efficient method for modeling exact and relaxed representations of the various nonoverlap and abutment constraints in combination with methods for maintaining the greatest possible macro-block mobility throughout all stages of optimization process.

To model the nonoverlap and abutment constraints we will build on the analytical computational-geometry methods developed in [10] and [11]. Although the recent work on electrostatics-based methods of [12, 13, 14] can be used to model a limited subset of the nonoverlap constraints needed for our work, this approach lacks the expressiveness, flexibility, high-level abstractions and software ecosystem needed to quickly and efficiently explore and develop the methods presented in this paper.

In this paper we will incrementally develop our methods using a sequence of three, successively more complex, floorplanning problems. The foundation for our methods will be developed by solving the first and simplest floorplanning problem; floorplanning with soft macro-blocks where, as shown in *Figure 1*, each macro-block is defined by a single nonoverlap-boundary, its physical-boundary, where all pins are assumed to be located at a single fixed point at the center of the physical-boundary.

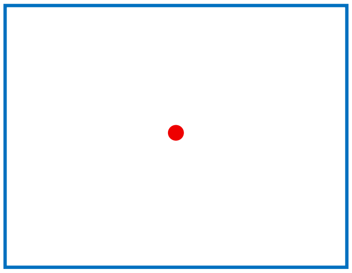

Figure 1: Spatial relationships between a macro-block’s physical-boundary (blue) and its centrally-located fixed-pin (red).

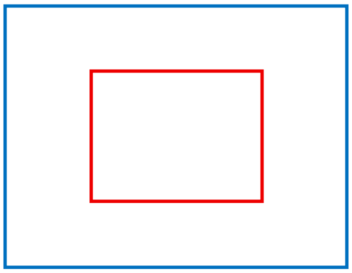

Figure 2: Spatial relationships between a macro-block's physical-boundary (blue) and its dynamic-pin containment-boundary (red).

For the second problem we will loosen the restrictions on the location of the macro-block pin. That is, we will extend the methods from the first problem to work for soft macro-blocks where, as shown in *Figure 2*, each macro-block consists of a single nonoverlap-boundary and a dynamic-pin containment-boundary inside of which the macro-block's pin is allowed to move.

The dynamic-pin containment-boundary is an axis-aligned rectangle, centered on the macro-block's physical-boundary, having the same aspect ratio as the physical-boundary and will be half its size. Within this containment-boundary, the optimum set of locations (see *Section 3.1.2*) for the macro-block's pin will be determined as part of the floorplanning process. Note that we are again, for demonstration purposes, simplifying the macro-block's pin structure. Ideally each of the macro-block's pins would move independent of each other within their own customized boundaries.

Although this paper deals exclusively with soft-macro floorplanning, it should be noted that this capability, when combined with hard-macros having multiple pin configurations (mirroring), provides the foundational capability needed by the floorplanner to select a particular pin configuration. That is, it can be used to generate an ideal location for each of a hard-macro's pins from which a metric can be constructed to determine which of that hard-macro's various pin configurations should be selected by the floorplanner.

For the third, and final, problem we will extend the methods from the second problem where, as shown in *Figure 3*, each macro-block now has a second nonoverlap-boundary, its thermal-boundary, which represents an abstraction of its "thermal signature". With this new boundary we can determine how close any two macros can approach based on the interaction of their thermal boundaries. In particular, the floorplanning process will, in addition to the basic requirement that no two physical-boundaries overlap, now select macro-block positions and aspect ratios such that no two thermal-boundaries overlap.

For the purposes of this paper, we represent each macro-block's thermal signature using a single thermal-boundary. However, it is not hard to see that this signature could be broken into multiple boundaries (for example, one which represents the macro-blocks heat-generation and another which represents its thermal-sensitivity) giving the floorplanner a more refined thermal-awareness.

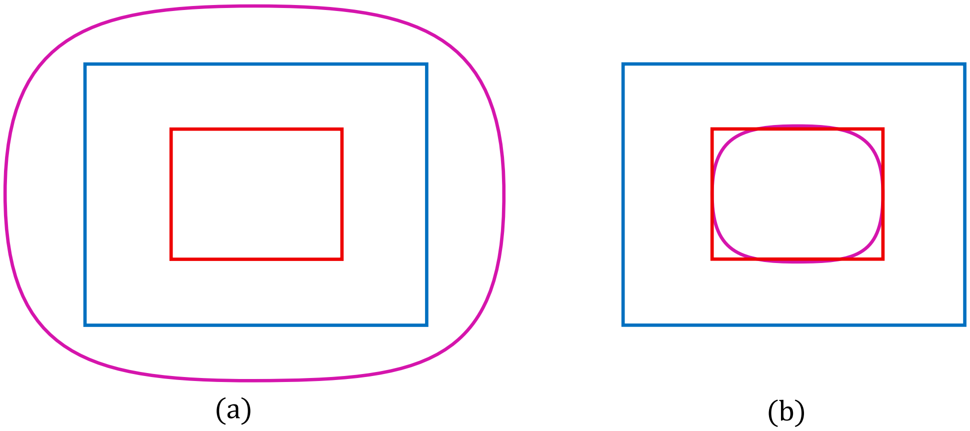

Figure 3: Spatial relationships between a macro-block's physical-boundary (blue), its thermal-boundary (magenta), and its dynamic-pin containment-boundary (red) for a thermal-boundary that is (a) *1.5X* its physical-boundary and for a thermal-boundary that is (b) *0.5X* its physical-boundary.

While, in practice, each macro-block's thermal boundary can take on a complex macro-dependent shape, we will, for the purposes of this paper and without loss of generality, model each thermal-boundary as a cubic superellipse having the same aspect ratio as the physical-boundary and a size that is between *0.5X* (*Figure 3*(b)) and *1.5X* (*Figure* ***3***(a)) the size of the physical-boundary.

Additionally, we will note that there are a number of additional considerations that can be formulated and incorporated into the floorplanning methods developed in this paper, however, since a full exploration of these additional considerations is beyond the scope of what can be reasonably developed in one paper, this paper will focus on the core techniques needed for implementing floorplanning methods which minimize total half-perimeter wire-length (*THPWL*) while satisfying the physical and thermal boundary constraints. Further, while the methods presented in this paper can be extended to solve both large-scale floorplanning, using active set (see, for example, [15] Section II.D and Fig. 4) and Nesterov accelerated-gradient methods, as well as hierarchical floorplanning (see [10] Section 8.4) both of these extensions are, similarly, beyond the scope of this paper.

Finally, in order to facilitate visual clarity when illustrating the results of our methods, we have limited the size (the number of macro-blocks, nets, and IO-bocks) of our floorplan examples. In particular, we have limited the number of nets to a quantity that makes it easier to visually evaluate relative improvements in net congestion through the various stages of the floorplanning methods.

The remainder of this paper is broken down into three parts. In Part 1 (*Section 2*) we will develop the solution to the Fixed-pin Single Nonoverlap-boundary Floorplanning problem. In Part 2 (*Sections 3*) we will build on the work of Part 1 to develop a solution to the Dynamic-pin Single Nonoverlap-boundary (Single-boundary) Floorplanning problem. And finally, in Part 3 (Section 4) we will extend the work of Part 2 to develop a solution to the Dynamic-pin Multiple Nonoverlap-boundary (Multi-boundary) Floorplanning problem.

Contents

1 Introduction.. 1

2 Fixed-Pin Single-Boundary Floorplanning............................... 7

2.1 Simplified Floorplanning.. 7

2.1.1 Objective Function.. 8

2.1.2 Area and Aspect-Ratio Constraints................................. 9

2.1.3 Physical-Boundary Nonoverlap Constraints.......................... 9

2.1.4 Fixed-Pin Containment Constraints................................. 11

2.1.5 Initial Layout.. 11

2.1.6 Simplified Floorplanning with *1X* Initial Spacing................ 16

2.1.7 Simplified Floorplanning with *10X* Initial Spacing............... 17

2.1.8 Relaxed Physical-Boundary Nonoverlap Constraints.................. 19

2.1.9 Simplified Floorplanning with Relaxed Boundaries.................. 19

2.2 Final Floorplanning... 23

2.2.1 Physical-Boundary Containment Constraints......................... 25

2.2.2 I/O-Pin Containment Constraints................................... 25

2.2.3 I/O-Block Nonoverlap Constraints.................................. 26

2.2.4 I/O-Block Abutment Constraints.................................... 26

2.2.5 Initial Layout.. 26

2.2.6 Final Floorplanning with Relaxed Boundaries....................... 29

2.2.7 Legalized Final-Floorplan... 31

3 Dynamic-Pin Single-Boundary Floorplanning............................. 32

3.1 Simplified Floorplanning.. 33

3.1.1 Dynamic-Pin Containment Constraints............................... 34

3.1.2 Simplified Floorplanning with Relaxed Boundaries.................. 35

3.2 Final Floorplanning... 38

3.2.1 Initial Layout.. 39

3.2.2 Final Floorplan with Relaxed Boundaries........................... 41

3.2.3 Legalized Final-Floorplan... 43

4 Dynamic-Pin Multi-Boundary Floorplanning.............................. 45

4.1 Simplified Floorplanning.. 45

4.1.1 Thermal-Boundary Nonoverlap Constraints........................... 46

4.1.2 Simplified Floorplanning with Relaxed Boundaries.................. 48

4.2 Final Floorplanning... 49

4.2.1 Final Floorplan with Relaxed Boundaries........................... 52

4.2.2 Legalized Final-Floorplan... 52

5 Conclusion.. 54

Figures

Figure 1: Spatial relationships between a macro-block's physical-boundary (blue) and its centrally-located fixed-pin (red). 2

Figure 2: Spatial relationships between a macro-block's physical-boundary (blue) and its dynamic-pin containment-boundary (red). 3

Figure 3: Spatial relationships between a macro-block's physical-boundary (blue), its thermal-boundary (magenta), and its dynamic-pin containment-boundary (red) for a thermal-boundary that is (a) *1.5X* its physical-boundary and for a thermal-boundary that is (b) *0.5X* its physical-boundary. 4

Figure 4: Spatial relationships for fixed-pin containment constraints................. 10

Figure 5: Initial layout for $m = 49$ soft macro-block physical-boundaries with 1:1 aspect ratios and *1X* spacing. 13

Figure 6: Initial net bounding-boxes for each of $n = 21$ nets. 14

Figure 7: Minimum bounding box enclosing all of the pins for net 1.................... 15

Figure 8: Fixed-pin Single-boundary Simplified-floorplan starting from *1X* initial spacing. 16

Figure 9: Initial layout of $m = 49$ soft macros with 1:1 aspect ratios and *10X* spacing.. 17

Figure 10: Fixed-pin Single-boundary Simplified-floorplan starting from *10X* initial spacing. 18

Figure 11: Fixed-pin Single-boundary Simplified-floorplan starting from *10X* initial spacing and relaxed macro-block physical-boundaries. 20

Figure 12: Rectangular macro-block boundary overlaps in a simplified-floorplan generated using relaxed macro-block physical-boundaries and *10X* initial spacing........... 21

Figure 13: Legalized floorplan derived from the simplified-floorplan generated using relaxed macro-block physical-boundaries and *10X* initial spacing................ 22

Figure 14: Spatial relationships for macro-block containment-constraints............... 25

Figure 15: Fixed-pin Single-boundary Compacted Simplified-floorplan with 10 relaxed I/O-block boundaries. 29

Figure 16: Fixed-pin Single-boundary Final-floorplan with relaxed boundaries........... 30

Figure 17: Fixed-pin Single-boundary Final-floorplan with rectangular boundaries....... 32

Figure 18: Spatial relationships for dynamic-pin containment constraints.............. 34

Figure 19: Dynamic-pin Single-boundary Simplified-floorplan starting from *10X* initial spacing and relaxed macro-block physical-boundaries. 37

Figure 20: Dynamic-pin Single-boundary Compacted Simplified-floorplan with 10 relaxed I/O-block boundaries. 42

Figure 21: Dynamic-pin Single-boundary Final-floorplan with relaxed boundaries......... 43

Figure 22: Dynamic-pin Single-boundary Final-floorplan with rectangular boundaries..... 44

Figure 23: Dynamic-pin Multi-boundary Simplified-floorplan starting from *10X* initial spacing and relaxed macro-block physical-boundaries. 48

Figure 24: Dynamic-pin Multi-boundary Compacted Simplified-floorplan with 10 relaxed I/O-block boundaries. 51

Figure 25: Dynamic-pin Multi-boundary Final-floorplan with relaxed boundaries.......... 52

Figure 26: Dynamic-pin Multi-boundary Final-floorplan with rectangular physical-boundaries. 53

2 Fixed-Pin Single-Boundary Floorplanning

In this section we will develop a solution for determining the location and aspect ratio of a set of soft macro-blocks with fixed-pins and a single nonoverlap-boundary, where the boundary corresponds to the macro-block's physical-size (its physical boundary). Additionally, we will determine the location of a set of fixed-size I/O-blocks and the dimensions of the floorplan boundary.

Due to the macro-block physical-boundary nonoverlap-constraints and I/O-block nonoverlap-constraints, the floorplanning problem is a nonconvex optimization-problem. Because it is nonconvex, the initial values of the optimization problem's optimization variables, from which the optimizer will begin the optimization process, can have a significant impact on the quality of the resulting floorplan. In particular, we have observed that this optimization problem can be made (see *section 2.1.5* through *2.1.9*) relatively insensitive to the initial macro-block positions, while at the same time, it is quite sensitive to the initial floorplan-boundary shape and I/O-block positions. This sensitivity is due to two primary mechanisms. First, I/O block placement is fundamentally a one-dimensional packing problem which has been "wrapped around" the floorplan-boundary. This gives the optimizer fewer degrees of freedom to solve it and, consequently, the I/O blocks have an outsized impact on the shape and position of the nets to which they belong. Specifically, due to the I/O blocks limited mobility, each I/O block's net will, necessarily, be dragged towards whichever side of the floorplan the I/O block resides. Second, the I/O blocks interaction with the floorplan-boundary act to compact the floorplan during the optimization process. This floorplan-compaction process systematically and prematurely reduces macro-block mobility limiting the optimizer's ability to cluster macro-blocks belonging to the same net.

As a result, we will solve the floorplanning problem using a two-stage successive-refinement strategy. In the first stage, the simplified-floorplanning stage, we will solve the floorplanning problem in the absence of floorplan-boundary and I/O-block information. That is, the simplified-floorplanning stage will generate the location and aspect-ratio of the macro-blocks entirely from the interconnections between the macro-blocks and the initial macro-block positions. The second stage, the final-floorplanning stage, will then generate the complete floorplan by refining the simplified-floorplan using an analytical optimization which now includes the floorplan-boundary and I/O-block information. We will initialize this stage using the macro-block position and aspect ratios of the simplified-floorplan and derive the initial floorplan-boundary shape and initial position for each of the I/O-blocks from information extracted from the simplified-floorplan.

2.1 Simplified Floorplanning

A formal description of the Simplified Fixed-pin Single-boundary Floorplanning problem can be stated as follows:

Given:

- A set, M, containing m macro-block physical-boundaries where each macro-block physical-boundary, i, is an axis-aligned rectangle whose origin, (mx_i, my_i), located at the center of the rectangle and has a height mh_i, width mw_i, and a fixed area ma_i.

- A set, N, containing n net bounding-boxes where each of the net bounding-boxes, j, is an axis-aligned rectangle with its origin, (nx_j, ny_j), located at the center of the rectangle and has a height nh_j, and width nw_j.
- Associated with each of the n net bounding-boxes in N, a set of macro-block pins, P_j, containing p_j pins where each of the pins, k, is located at $(px_{j,k}, py_{j,k})$ which, for the fixed-pin case, is the origin of its corresponding macro-block, (mx_i, my_i).

Determine:

- The location of the origin, (mx_i, my_i), the height, mh_i, and the width, mw_i, for each of the m physical-boundaries in M.
- The location of the origin, (nx_j, ny_j), the height, nh_j, and the width, nw_j, for each of the n net bounding-boxes in N.

Such that:

- The total half-perimeter wire-length of the n net bounding-boxes, in N, is minimized.
- No two physical-boundaries in M overlap.
- The aspect ratio for each of the m physical-boundaries is between 0.5 and 2.
- The area of each of the m physical boundaries is ma_i
- For each of the n nets, j, each pin, k, in P_j is contained in net j's bounding-box.

From this, we can formulate the Simplified Fixed-pin Single-boundary Floorplanning problem as the following high-level optimization:

Minimize Total Half-Perimeter Wire-length

while satisfying the following four sets of constraints:

1. Nonoverlap constraints between all pairs of physical-boundaries in M.
2. Physical-boundary aspect ratio constraints.
3. Physical-boundary area constraints.
4. Containment constraints to keep each fixed-pin contained in the bounding-box of each of its associated nets.

2.1.1 Objective Function

With the exception of the floorplan-compaction problems (see *Sections 2.2.5* and *3.2.1*), the objective function for all floorplanning problems presented in this paper are the same. Specifically, the total half-perimeter wire-length (*THPWL*), which is formulated as:

$$THPWL = \sum_{j=1}^{n} \left(nh_j + nw_j\right) \tag{1}$$

Note that we are using an exact calculation of *THPWL* as opposed to a Naylor log-sum-exponent approximation [16] used in many analytical placement and floorplanning implementations. This approach gives us a simple linear-objective, with none of the differentiability issues associated with the use of the *max* operation, at the expense of an added set of linear pin-containment constraints developed in *Section 2.1.4*. For the floorplanning problem this increase in constraint count is, in most cases, a reasonable tradeoff for the improved accuracy and simplicity of the *THPWL* calculation.

2.1.2 Area and Aspect-Ratio Constraints

The area constraints (constraint 3) for a macro-block, i, can be formulated as:

$$mh_i \cdot mw_i = ma_i \tag{2}$$

while the aspect ratio constraints (constraint 2) can be formulated as:

$$mh_i / mw_i \leq 2 \tag{3}$$

$$mh_i / mw_i \geq 0.5 \tag{4}$$

Substituting for mw_i we have:

$$mh_i^2 / ma_i \leq 2 \tag{5}$$

$$mh_i^2 / ma_i \geq 0.5 \tag{6}$$

Rearranging produces the following pair of bounds on mh_i:

$$mh_i \leq \sqrt[2]{2\,ma_i} \tag{7}$$

$$mh_i \geq \sqrt[2]{0.5\,ma_i} \tag{8}$$

Although this paper deals exclusively with soft-macro floorplanning, it should be noted that by replacing this pair of bound constraints with the following equality constraint on the macro's half-perimeter:

$$mh_i + mw_i = \sqrt[2]{ma_i / R} + \sqrt[2]{R \cdot ma_i} \tag{9}$$

the optimizer can be made to select the orientation of a hard-macro. In particular, if a hard-macro's orientation is represented by its aspect ratio, 1:R or R:1, equations (2) and (9) form a system of equations in mh_i and mw_i having two solutions, one giving us a macro-block with aspect ratio 1:R and the other an aspect ratio R:1.

2.1.3 Physical-Boundary Nonoverlap Constraints

To construct the macro-block physical-boundary nonoverlap constraints (constraint 1) we will formulate each nonoverlap constraint using an ACGL [10, 17] bound-shape-function as the constraint function. As outlined in *Section 3* of [11], two shapes do NOT overlap when the bound-shape-function of their closest-approach shape-boundary is greater than one.

Specifically, given two macro-blocks, M_1 and M_2, from the set M, we can form two rectangular shape-boundaries, RB_1 and RB_2, as:

$$RB_1 = RectangleBoundary((DiffVar)\,mw_1,\ (DiffVar)\,mh_1) \tag{10}$$

$$RB_2 = RectangleBoundary((DiffVar)\,mw_2,\ (DiffVar)\,mh_2) \tag{11}$$

Using RB_1, RB_2 and the position of their origins, (mx_1, my_1) and (mx_2, my_2), we can form the bound-shape-function, BSF, of their closest-approach shape-boundary as:

$$BSF = BoundShapeFunction(ClosestApproachBoundary(RB_1, RB_2), \\ (DiffVar)mx_1, (DiffVar)my_1, (DiffVar)mx_2, (DiffVar)my_2) \quad (12)$$

Using the *BoundShapeFunction*'s *evaluateFunct* method, we can formulate the nonoverlap-constraint between the physical-boundaries of M_1 and M_2 as:

$$BSF.evaluteFunct() \geq 1 \quad (13)$$

Further, in order for IPOPT to efficiently find solutions we will need to provide it with the Jacobian matrix of the optimization problem's constraint functions. Consequently, we need to extract the gradient of this nonoverlap-constraint with respect to its differentiable variables mw_1, mh_1, mw_2, mh_2, mx_1, my_1, mx_2, and my_2. Using the *BoundShapeFunction*'s *evaluateGrad* method, the gradient can be extracted as follows:

$$GV = BSF.evaluateGrad() \quad (14)$$

where GV is the vector of partial derivatives.

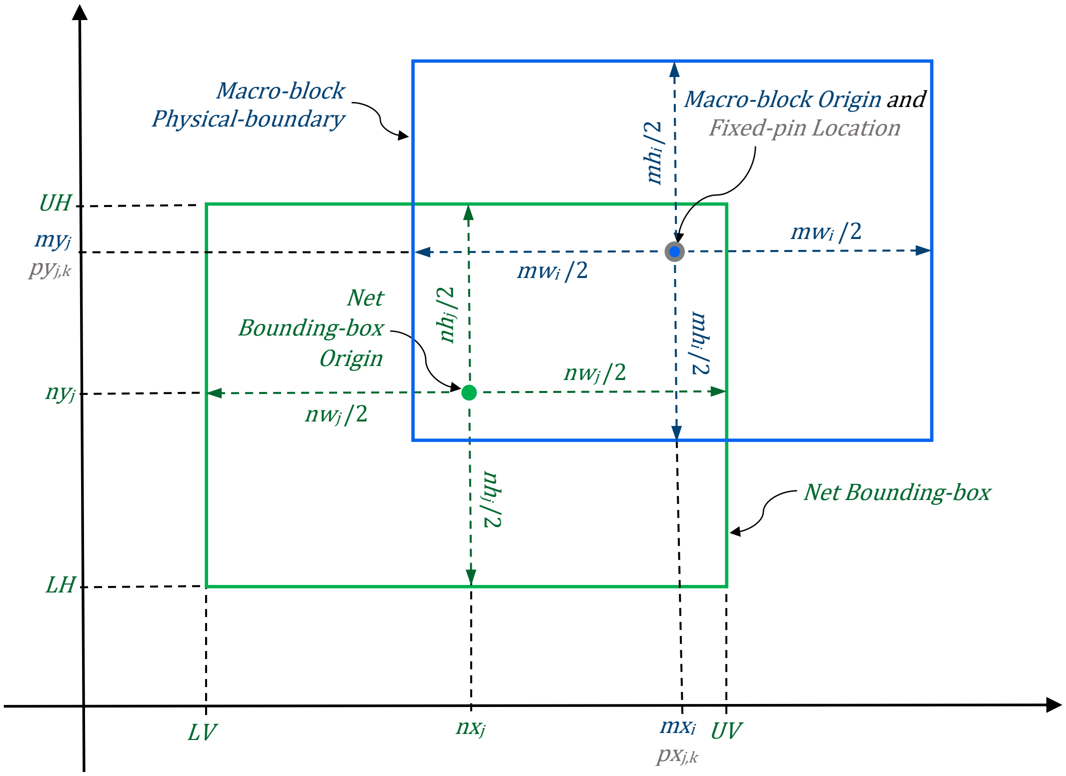

Figure 4: Spatial relationships for fixed-pin containment constraints.

2.1.4 Fixed-Pin Containment Constraints

To formulate the fixed-pin containment constraints (constraint 4), we begin by observing that, as shown in *Figure 4*, in order for a fixed-pin to be contained inside the net bounding-box, the pin must be to the left of UV, to the right of LV, below UH and above LH. For a pin k associated with macro-block i and net j, these requirements can be formulated as:

$$px_{j,k} \leq UV \tag{15}$$

$$px_{j,k} \geq LV \tag{16}$$

$$py_{j,k} \leq UH \tag{17}$$

$$py_{j,k} \geq LH \tag{18}$$

Rewriting UV, UH, LV, and LH as:

$$UV = nx_j + nw_j/2 \tag{19}$$

$$LV = nx_j - nw_j/2 \tag{20}$$

$$UH = ny_j + nh_j/2 \tag{21}$$

$$LH = ny_j - nh_j/2 \tag{22}$$

and since, for the fixed-pin case, each pin is located at the origin of its corresponding macro-block, the fixed-pin containment-constraints can be reformulated as:

$$mx_i \leq nx_j + nw_j/2 \tag{23}$$

$$mx_i \geq nx_j - nw_j/2 \tag{24}$$

$$my_i \leq ny_j + nh_j/2 \tag{25}$$

$$my_i \geq ny_j - nh_j/2 \tag{26}$$

2.1.5 Initial Layout

Due to its nonoverlap constraints, the floorplanning problem is a nonconvex optimization problem. Because of this, the initial values of the optimization problem's optimization variables, from which the optimizer will begin the optimization process, can have a large impact on the optimization results. For this reason, we need to choose initial values for the origin, width, and height (mx_i, my_i, mw_i, mh_i) of each of the m macro-block physical-boundaries. Additionally, we need to choose initial values for the origin, width, and height (nx_j, ny_j, nw_j, nh_j) of each of the n net bounding-boxes.

As shown in *Figure 5*, for a design containing $m = 49$ macro-blocks, we select the initial width and height of each macro-block so that they have a 1:1 aspect ratio, that is we set mw_i and mh_i as:

$$mh_i = mw_i = \sqrt[2]{ma_i} \tag{27}$$

or, using a numerically stable approach:

$$mw_i = \sqrt[2]{ma_i} \tag{28}$$

$$mh_i = ma_i/mw_i \tag{29}$$

The origin of each macro-block, (mx_i, my_i), is selected such that each block is placed into a square cell of a square grid where the area of every cell is equal to the area of the largest of the 49 macro-block physical-boundaries, $\max_{i=1,m}(ma_i)$. We will refer to this initial layout as an initial layout with *1X* spacing.

Note that each macro-block physical-boundary in *Figure 5* is shown as a blue rectangle labeled with an “M” followed by its macro-block number. In this case, M1 through M49. Further, the 3rd row of macro-blocks (macro-blocks 15 through 21) has been shaded to highlight their original position and make it easier to evaluate their mobility, as inferred by the relative change in their positions in subsequent steps of the floorplanning process. Finaly, each macro-block pin is shown as a red circle at the center of its macro-block physical-boundary.

Figure 6 shows the initial net bounding-boxes for each of $n = 21$ nets. Each net’s origin, height, and width (nx_j, ny_j, nw_j, nh_j) are selected such that they form the minimum sized bounding-box enclosing all of that net’s pins.

The initial total half-perimeter wire-length is 296. Note that the units of total half-perimeter wire-length are irrelevant as we are only interested in using this number to compute the relative change in total half-perimeter wire-length achieved by subsequent steps in the floor planning process.

Note that, as shown for *Net 1* in *Figure 7*, each net’s bounding box is drawn as a green box with its associated diagonals and is labeled with an “N” followed by its net number.

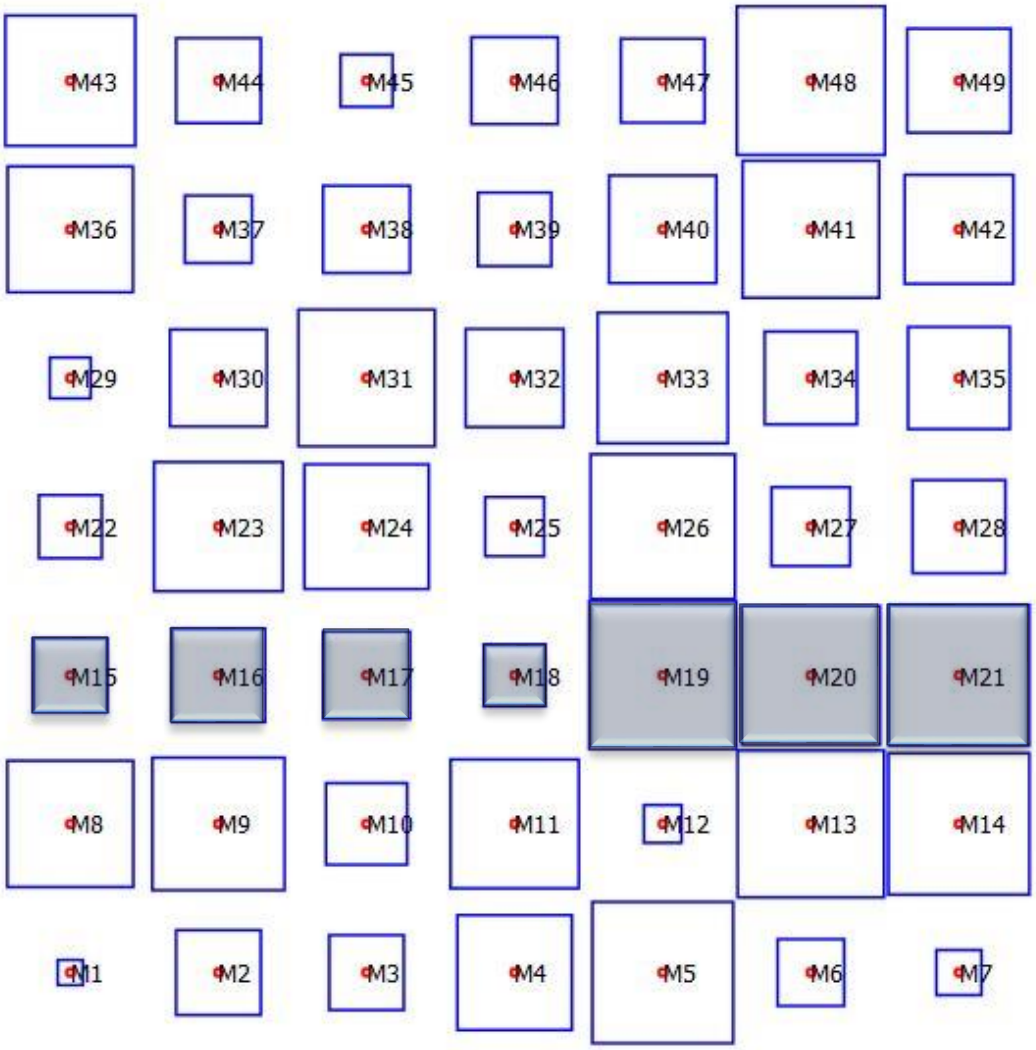

Figure 5: Initial layout for $m = 49$ soft macro-block physical-boundaries with 1:1 aspect ratios and *1X* spacing.

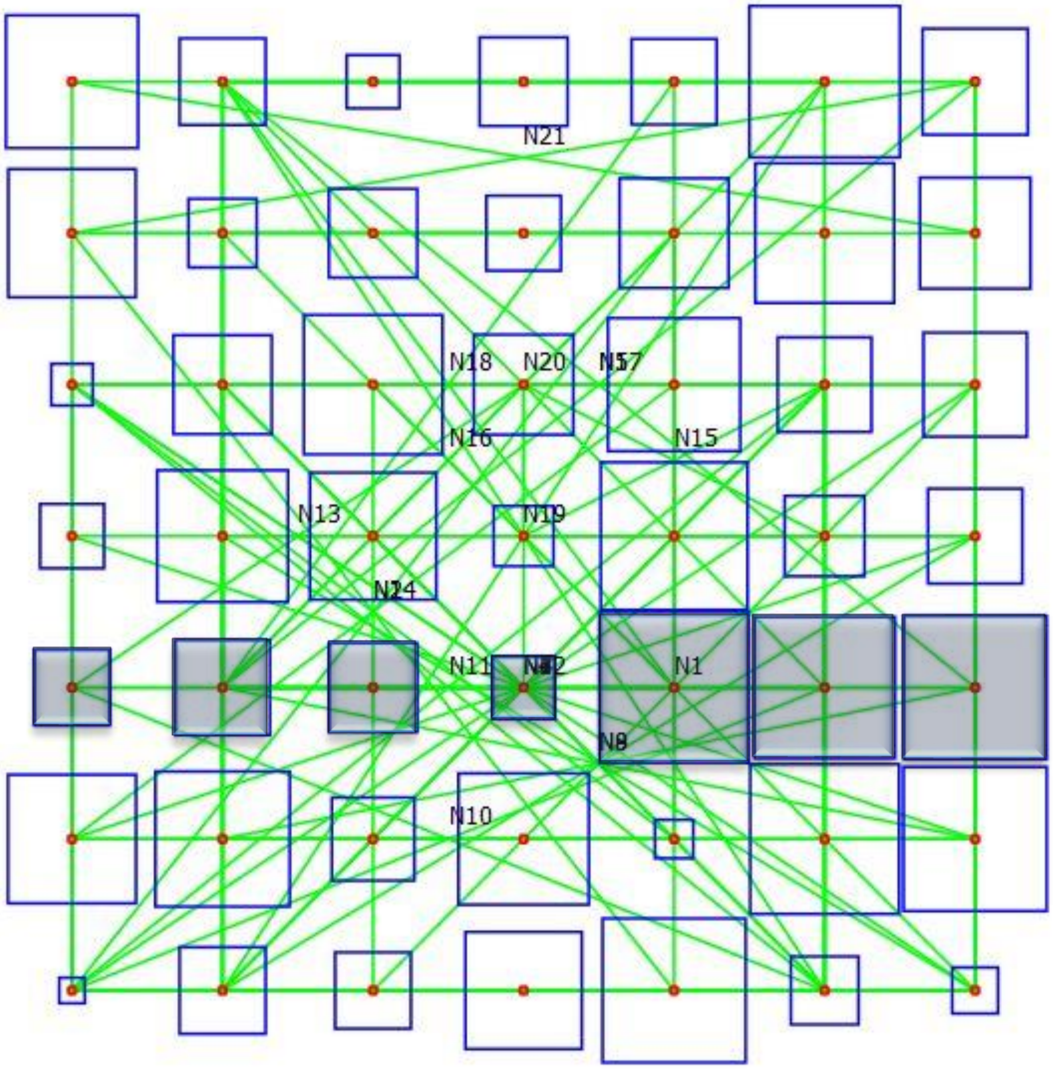

Figure 6: Initial net bounding-boxes for each of $n = 21$ nets.

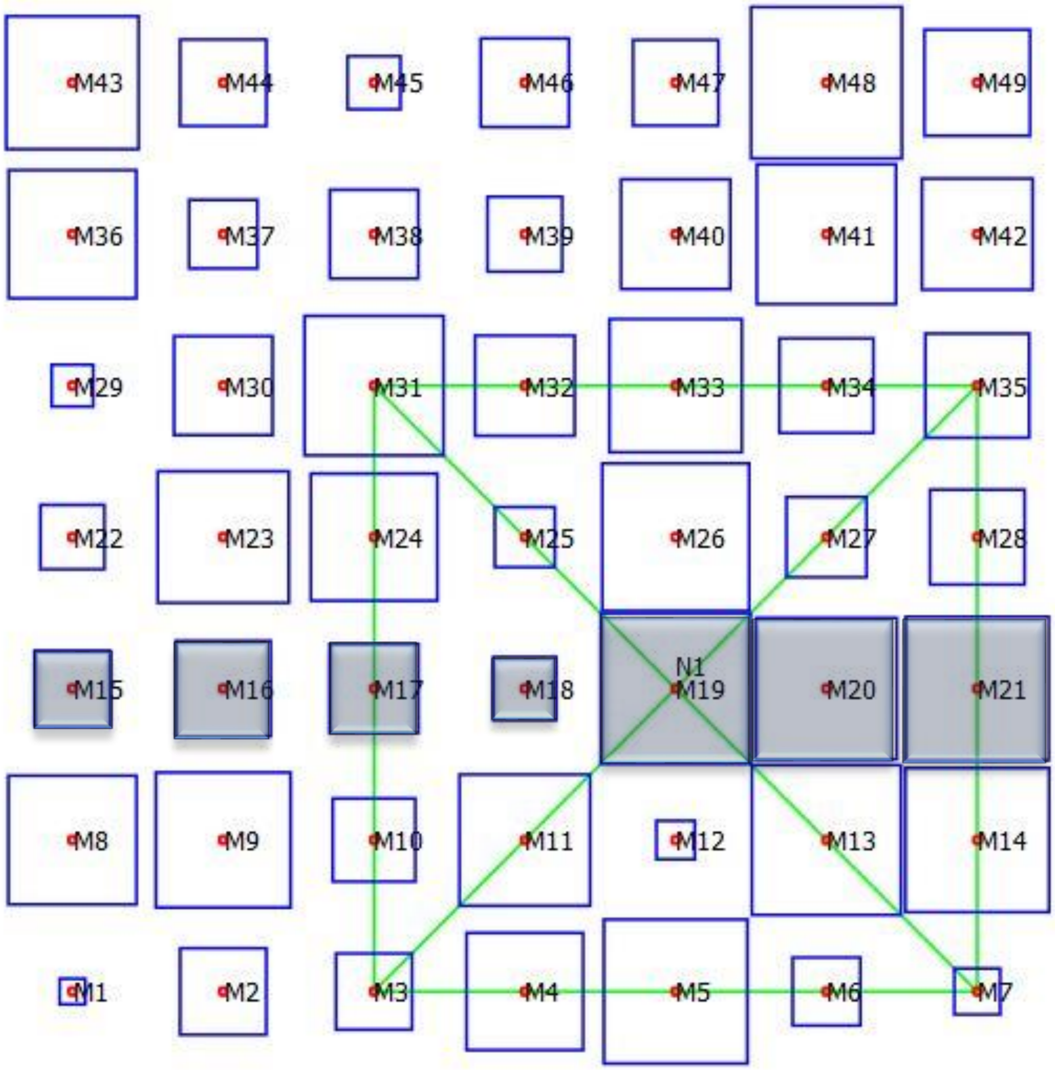

Figure 7: Minimum bounding box enclosing all of the pins for *Net* 1.

2.1.6 Simplified Floorplanning with *1X* Initial Spacing

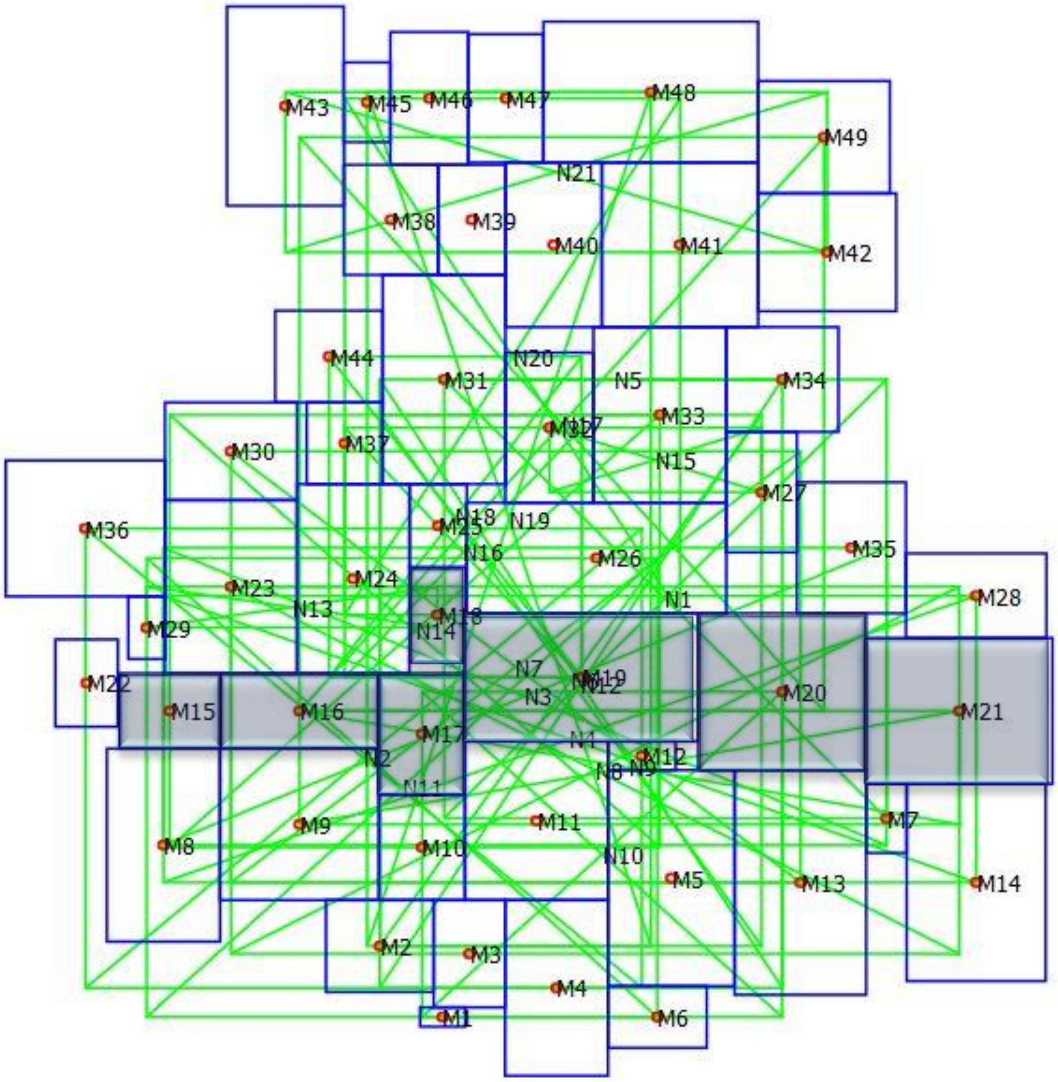

Figure 8: Fixed-pin Single-boundary Simplified-floorplan starting from *1X* initial spacing.

Solving the Fixed-pin Single-boundary Simplified-floorplanning problem with *1X* spacing using IPOPT produces the floorplan shown in *Figure 8*. The total half-perimeter wire-length of this floorplan is 188.6. While this is a 36.3% reduction over the initial total half-perimeter wire-length of the initial layout shown in *Figure 6*, the resulting floorplan is little more than a compaction of the macro-blocks as can be seen by the poor mobility and minimal change in the relative position of macro blocks 15 through 21. Additionally, we can see that little progress has been made in reducing net-congestion (the degree to which the nets are still overlapping). One of the primary causes for this minimal improvement is the fact that the macro-blocks start out in such close proximity. This in combination with the nonoverlap constraints gives the optimizer little room to move the macros past each other during an optimization process which is trying to tightly cluster macros which belong to the same net. This is essentially a distributed compaction process being performed on an already relatively compact layout; effectively locking many of the macros in place.

2.1.7 Simplified Floorplanning with *10X* Initial Spacing

To increase macro-block mobility and give the optimizer “more room to work” we will increase the area of the cell into which each macros block is initially placed. Specifically, the area of each cell will be increased to 10 times the area of the largest macro-block physical-boundary (*10X* spacing), as shown in *Figure 9*.

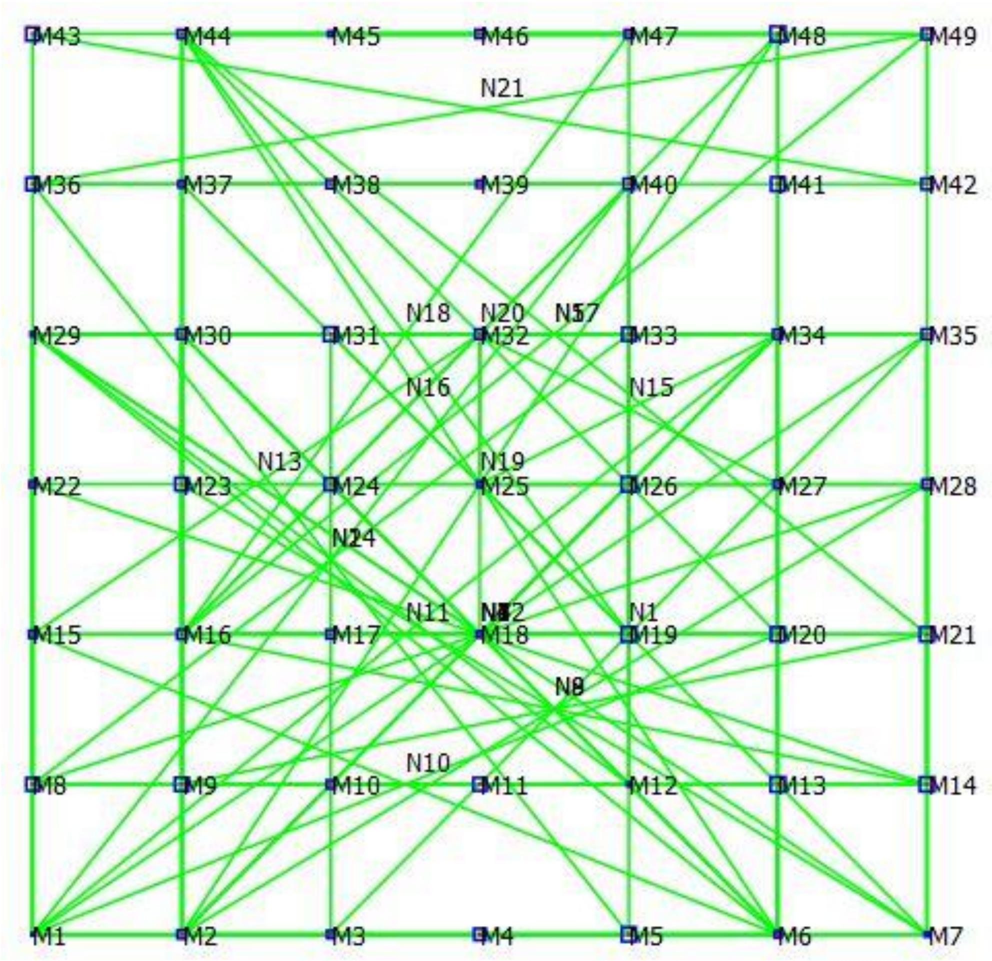

Figure 9: Initial layout of $m = 49$ soft macros with 1:1 aspect ratios and *10X* spacing.

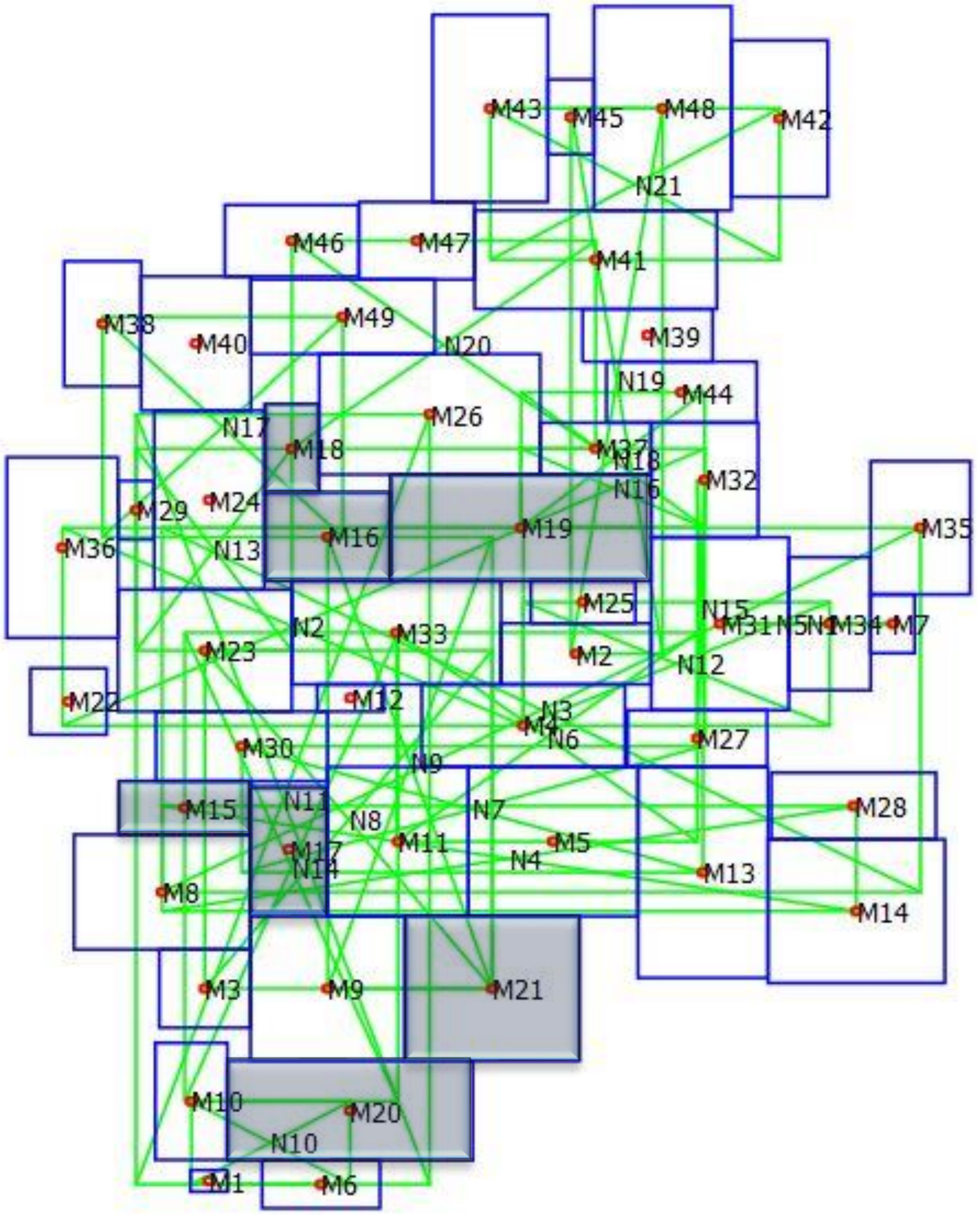

Figure 10: Fixed-pin Single-boundary Simplified-floorplan starting from *10X* initial spacing.

Solving the Fixed-pin Single-boundary Simplified-floorplanning problem with this *10X* initial spacing produces the floorplan shown in *Figure 10*. The total half-perimeter wire-length of this floorplan is 124.5, which is a 34% improvement over that achieved from the *1X* initial layout. While this result is a marked improvement in total half-perimeter wire-length, macro-block mobility, and net-congestion, we can see that there is still room for improvement in macro-block mobility as the macro-blocks are still “bunching up” and there is considerable room for improvement in net-congestion.

While we have improved “global” macro-mobility by increasing the initial spacing between macro-block physical-boundaries, “detailed” macro-mobility is being impeded by the adverse interaction between the axis-aligned rectangular-shape of the macro-block physical-boundaries and the compaction forces being applied by the axis-aligned net bounding-boxes. In particular, two axis-aligned rectangles cannot move past each other once they come into contact when the force pushing them together is also axis-aligned. Further, even when the force pushing two axis-aligned rectangles is not axis-aligned, the force transmitted through the system of rectangles will become axis-aligned as there is no friction between the two rectangles which can transmit the non-axis-aligned force. We can significantly reduce this impediment by eliminating the flat sides of the macro-block physical-boundaries and, instead, approximate them with curved sides. By doing this, any forces, axis-aligned or not, pushing the two blocks together will produce non-axis-aligned relative motion between the two blocks. For the purposes of this paper, we will approximate each macro-block's physical-boundary as a simple axis-aligned cubic-superellipse "relaxation" of its axis-aligned rectangular-shape.

2.1.8 Relaxed Physical-Boundary Nonoverlap Constraints

To construct the macro-block relaxed physical-boundary nonoverlap constraints we will again formulate each nonoverlap constraint using an ACGL bound-shape-function as the constraint function. Specifically, given two macro-blocks, M_1 and M_2, from the set M, we can form two cubic superellipse shape-boundaries, SEB_1 and SEB_2, as:

$$SEB_1 = \mathit{SuperellipseBoundary}(3.0,\ (\mathit{DiffVar})\, mw_1,\ (\mathit{DiffVar})\, mh_1) \tag{30}$$

$$SEB_2 = \mathit{SuperellipseBoundary}(3.0,\ (\mathit{DiffVar})\, mw_2,\ (\mathit{DiffVar})\, mh_2) \tag{31}$$

Using SEB_1, SEB_2 and the position of their origins, (mx_1, my_1) and (mx_2, my_2), we can form the bound-shape-function, BSF, of their closest-approach shape-boundary as:

$$\begin{aligned} BSF = \mathit{BoundShapeFunction}\,(&\mathit{ClosestApproachBoundary}(SEB_1,\ SEB_2), \\ &(\mathit{DiffVar})\, mx_1,\ (\mathit{DiffVar})\, my_1,\ (\mathit{DiffVar})\, mx_2,\ (\mathit{DiffVar})\, my_2) \end{aligned} \tag{32}$$

This allows us to, again, formulate the nonoverlap-constraint between the relaxed physical-boundaries of M_1 and M_2 as:

$$BSF.\mathit{evaluteFunct}() \geq 1 \tag{33}$$

and extract the gradient with respect to its differentiable variables mw_1, mh_1, mw_2, mh_2, mx_1, my_1, mx_2, and my_2 using:

$$GV = BSF.\mathit{evaluateGrad}() \tag{34}$$

where GV is the vector of partial derivatives for the bound-shape-function.

2.1.9 Simplified Floorplanning with Relaxed Boundaries

Solving the Fixed-pin Single-boundary Simplified-floorplanning problem with relaxed boundaries and a *10X* initial spacing produces the floorplan shown in *Figure 11*. The total half-perimeter wire-length of this floorplan is 88.4 which is a 29% improvement over that achieved from the *10X* initial layout with rectangular-shaped macro-blocks and a 53.1% improvement over that achieved from the *1X* initial layout with rectangular-shaped macro-blocks. Additionally, it shows dramatic improvement in net congestion over both the *1X* and *10X* rectangular-shaped macro-block cases as well as moderate improvement in macro-block mobility, as most of the bunching from the *10X* case has been alleviated. However, while the floorplan produced using the relaxed macro-bock boundaries has successfully eliminated the binding between macro-blocks, the actual rectangular shapes of the macro-blocks still have unresolved overlaps, as shown in *Figure 12*.

To eliminate these overlaps (legalize the simplified-floorplan) we re-solve the Simplified-floorplanning problem using the nonoverlap constraints constructed from the macro-block's rectangular boundaries (equations (10) through (13)) and initialize the optimization variables to the values produced by the simplified-floorplan with relaxed macro-block boundaries. Further, we solve this floorplan legalization problem using IPOPT's warm-start capability in order to ensure that the resulting legalized floorplan does not stray too far from the floorplan produced using the relaxed macro-block boundaries.

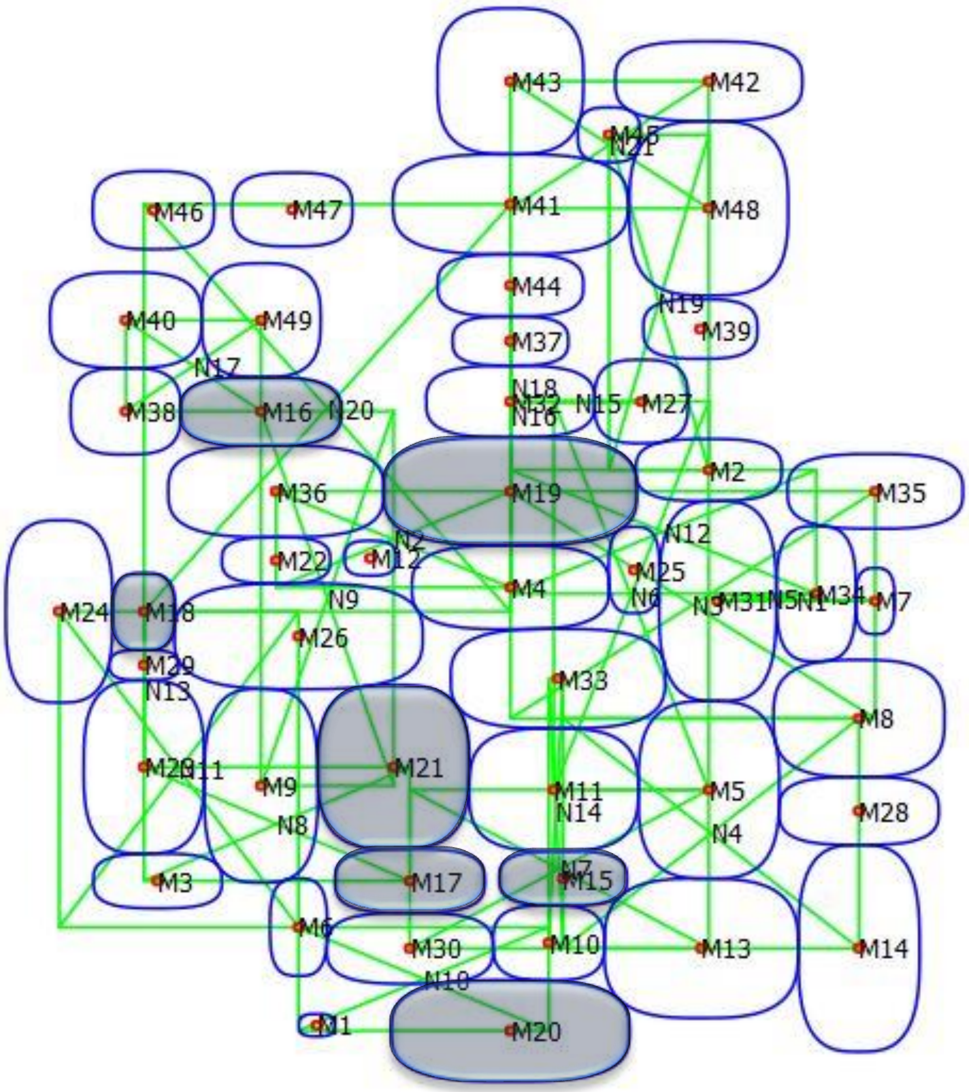

Figure 11: Fixed-pin Single-boundary Simplified-floorplan starting from *10X* initial spacing and relaxed macro-block physical-boundaries.

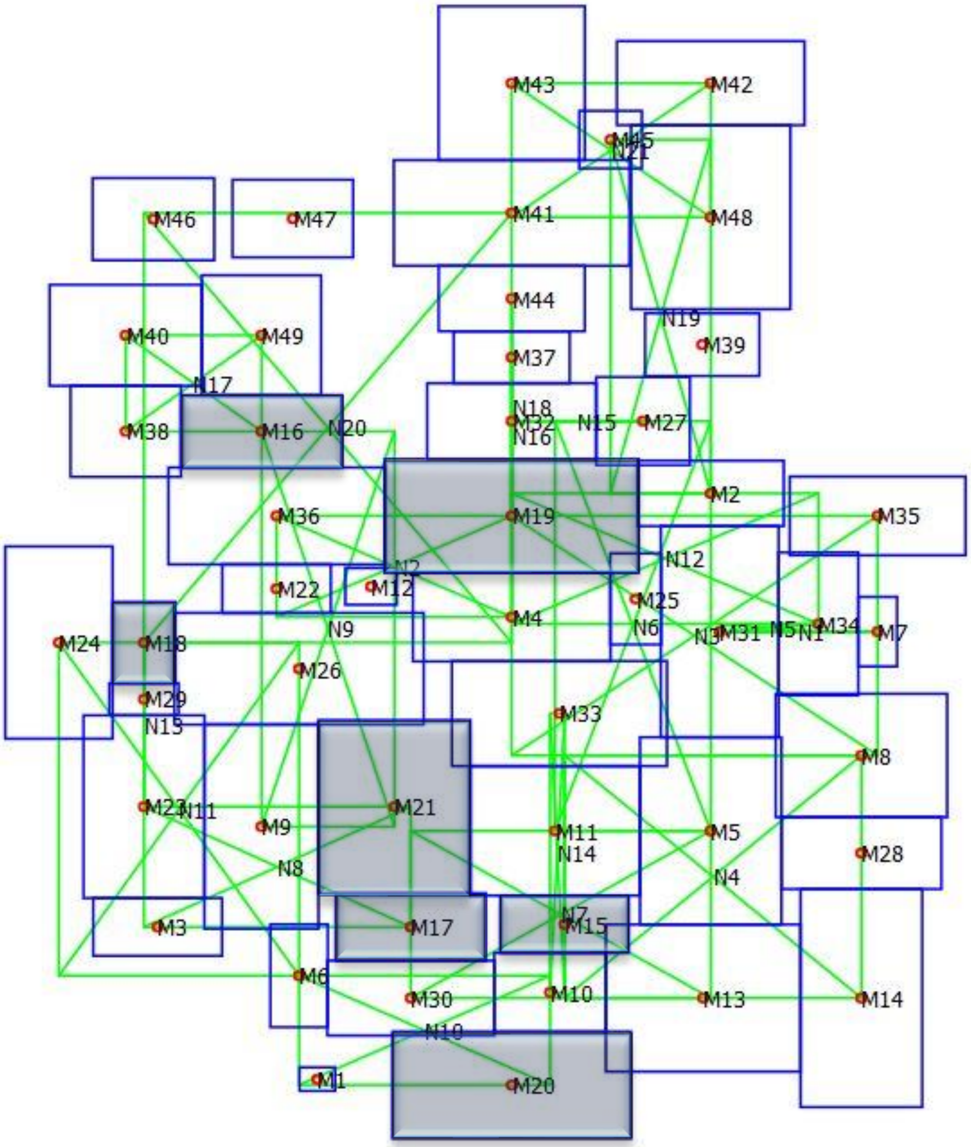

Figure 12: Rectangular macro-block boundary overlaps in a simplified-floorplan generated using relaxed macro-block physical-boundaries and *10X* initial spacing.

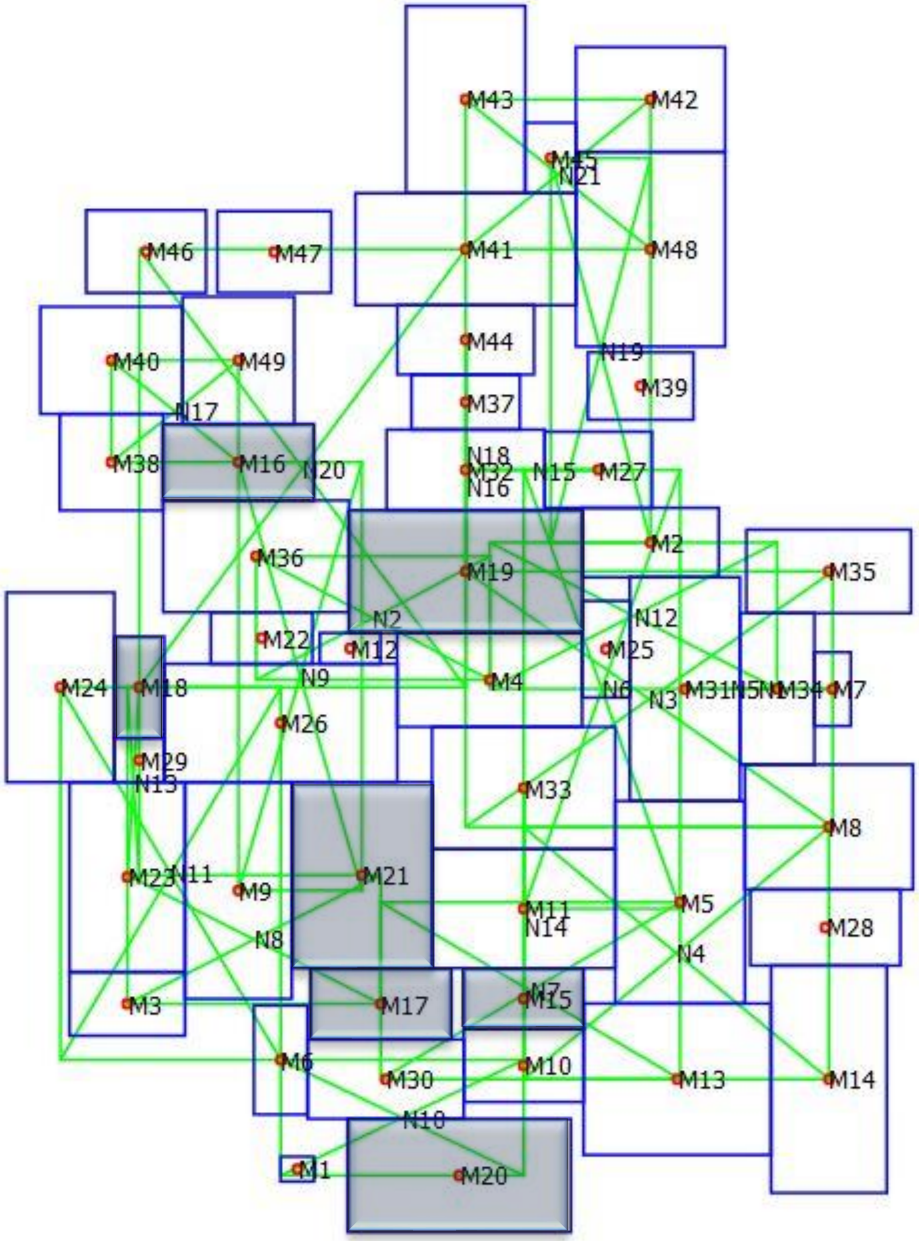

Figure 13: Legalized floorplan derived from the simplified-floorplan generated using relaxed macro-block physical-boundaries and *10X* initial spacing.

The legalized floorplan, shown in *Figure 13*, has a total half-perimeter wire-length of 90.1. This is a 2% increase in wire length from that produced using the relaxed macro-block boundaries (*Figure 11*) but still provides a 27.6% improvement over that produced from the *10X* initial spacing with rectangular macro-block boundaries (*Figure 10*) as well as a 52.2% improvement over that produced using a *1X* initial spacing (*Figure 8*) while still retaining the improvements in net congestion and macro-block mobility obtained using the relaxed macro-block boundaries (*Figure 11*).

2.2 Final Floorplanning

In the final-floorplanning stage we will use the information derived from the relaxed simplified-floorplan to provide the starting point for solving the full Fixed-pin Single-Boundary Floorplanning problem. This Fixed-pin Single-Boundary Floorplanning problem can be stated as follows:

Given:

- An axis-aligned rectangle which is a bounding-box representing the floorplan-boundary, FB, with its origin, (fbx, fby), located at the center of the rectangle, and having a height fbh, and width fbw.
- A set, M, containing m macro-block physical-boundaries where each macro-block physical-boundary, i, is an axis-aligned rectangle whose origin, (mx_i, my_i), located at the center of the rectangle and has a height mh_i, width mw_i, and a fixed area ma_i.
- A set, N, containing n net bounding-boxes where each of the net bounding-boxes, j, is an axis-aligned rectangle with its origin, (nx_j, ny_j), located at the center of the rectangle and has a height nh_j, and width nw_j.
- Associated with each of the n net bounding-boxes in N, a set of macro-block pins, P_j, containing p_j pins where each of the pins, k, is located at $(px_{j,k}, py_{j,k})$ which, for the fixed-pin case, is the origin of its corresponding macro-block, (mx_i, my_i).
- A set, IO, containing io I/O-block boundaries where each I/O-block boundary, q, is an axis-aligned square whose origin, (iox_q, ioy_q), located at the center of the square and all I/O-block boundaries have the same fixed height and width, $iohw$. Each of the io I/O-blocks in IO is associated with one of the n nets in N, ion_q, and no net can be associated with more than one I/O-block.

Determine:

- The location of the origin, (fbx, fby), the height, fbh, and the width, fbw, of the floorplan-boundary FB.
- The location of the origin, (mx_i, my_i), the height, mh_i, and the width, mw_i, for each of the m macro-block physical-boundaries in M.
- The location of the origin, (nx_j, ny_j), the height, nh_j, and the width, nw_j, for each of the n net bounding-boxes in N.
- The location of the origin, (iox_q, ioy_q), for each of the io fixed-size I/O-blocks in IO.

Such that:

- The total half-perimeter wire-length of the n net bounding-boxes, in N, is minimized.
- No two physical-boundaries in M overlap.
- The floorplan-boundary bounding-box, FB, encloses all of the m physical-boundaries in M.
- The aspect ratio for each of the m physical-boundaries in M is between 0.5 and 2.
- The area of each of the m physical-boundaries, i, in M is ma_i

- For each of the *n* nets, *j*, each pin, *k*, in P_j is contained in the bounding-box for net *j*.
- Each of the *io* I/O-blocks, *q*, in *IO* has its origin, (iox_q, ioy_q), contained in the bounding box for net ion_q.
- No two of the *io* I/O-block boundaries in *IO* overlap.
- Each of the *io* I/O-block boundaries abuts the floorplan-boundary bounding-box, *FB*.

From this, we can formulate the Final Fixed-pin Single-boundary Floorplanning problem as the following high-level optimization:

Minimize Total Half-Perimeter Wire-length

while satisfying the following eight sets of constraints:

1. Nonoverlap constraints between all pairs of physical-boundaries in *M*.
2. Physical-boundary aspect ratio constraints.
3. Physical-boundary area constraints.
4. Containment constraints to keep each fixed-pin contained in its associated net's bounding-box.
5. Containment constraints to keep each I/O-pins contained in it associated net's bounding-box.
6. Containment constraints to keep each physical-boundary contained in the floorplan-boundary.
7. Abutment constraints to abut each I/O-block to the floorplan-boundary.
8. Nonoverlap constrains between all pairs of I/O-block boundaries in *IO*.

We will solve this problem in two phases. In the first phase we will solve a relaxed version of this problem. That is, we will solve it using relaxed boundaries for the macro-blocks (as we did in *Section 2.1.9*) and relaxed boundaries for the I/O-blocks. For the macro-blocks we will, again, use cubic superellipses for their relaxed boundaries, and for the I/O-blocks we will use circles (with diameter *iohw*) as their relaxed boundaries. In the second phase we will legalize the relaxed solution (as we did in *Section 2.1.9*) to eliminate any unresolved macro-block and I/O-block overlaps.

2.2.1 Physical-Boundary Containment Constraints

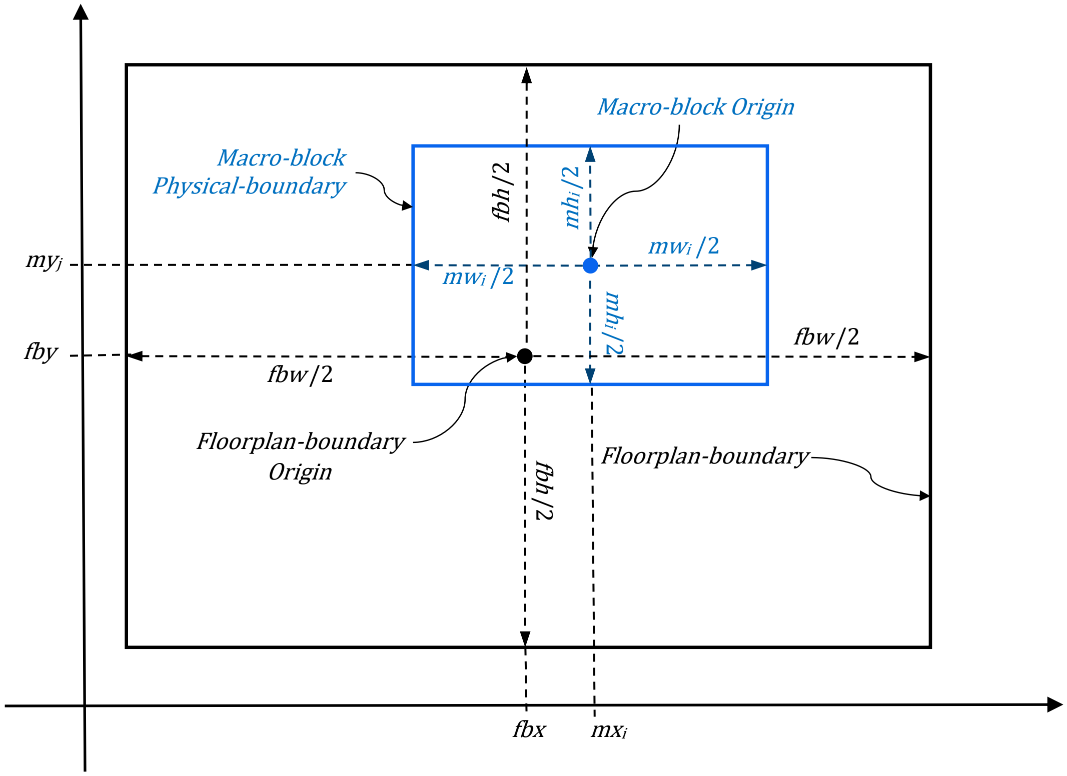

Figure 14: Spatial relationships for macro-block containment-constraints.

In order for a macro-block to be contained inside the floorplan-boundary (constraint 6), as shown in *Figure 14*, the left side of the macro-block must be to the right of the left side of the floorplan-boundary and the right side of the macro-block must be to the left of the right side of the floorplan-boundary. Similarly, the bottom of the macro-block must be above the bottom of the floorplan-boundary, and the top of the macro-block must be below the top of the floorplan-boundary. From this, the containment constraints for a macro-block, *i*, can be formulated as:

$$fbx - fbw/2 \leq mx_i - mw_i/2 \tag{35}$$

$$fbx + fbw/2 \geq mx_i + mw_i/2 \tag{36}$$

$$fby - fbh/2 \leq my_i - mh_i/2 \tag{37}$$

$$fby + fbh/2 \geq my_i + mh_i/2 \tag{38}$$

2.2.2 I/O-Pin Containment Constraints

Using the approach outlined in *Section 2.1.4*, the containment constraints (constraint 5) for an I/O-pin, *q*, can be formulated as:

$$iox_q \leq nx_{ion_q} + nw_{ion_q}/2 \tag{39}$$

$$iox_q \geq nx_{ion_q} - nw_{ion_q}/2 \tag{40}$$

$$ioy_q \leq ny_{ion_q} + nh_{ion_q}/2 \tag{41}$$

$$ioy_q \geq ny_{ion_q} - nh_{ion_q}/2 \tag{42}$$

2.2.3 I/O-Block Nonoverlap Constraints

To construct the I/O-block relaxed physical-boundary nonoverlap constraints (constraint 8) we will, again, formulate each nonoverlap constraint using an ACGL bound-shape-function as the constraint function. That is, given two I/O-blocks, IO_1 and IO_2, from the set IO, we form two circle shape-boundaries, CB_1 and CB_2, as:

$$CB_1 = CircleBoundary(iohw) \tag{43}$$

$$CB_2 = CircleBoundary(iohw) \tag{44}$$

Using CB_1, CB_2 and the position of their origins, (iox_1, ioy_1) and (iox_2, ioy_2), we can form the bound-shape-function, BSF, of their closest-approach shape-boundary as:

$$BSF = BoundShapeFunction(ClosestApproachBoundary(CB_1, CB_2), \\ (DiffVar)iox_1, (DiffVar)ioy_1, (DiffVar)iox_2, (DiffVar)ioy_2) \tag{45}$$

This allows us to formulate the nonoverlap-constraint between the relaxed-boundaries of IO_1 and IO_2 as:

$$BSF.evaluteFunct() \geq 1 \tag{46}$$

2.2.4 I/O-Block Abutment Constraints

To construct the I/O-block relaxed-boundary abutment-constraints (constraint 7) we will formulate each abutment-constraint using an ACGL bound-shape-function as the constraint function. Specifically, given an I/O-blocks, IO_1, from the set IO, and the floorplan-boundary, FB, we can form a circle and a rectangle shape-boundaries, CB and RB, as:

$$CB = CircleBoundary(iohw) \tag{47}$$

$$RB = RectangleBoundary((DiffVar)fbw, (DiffVar)fbh) \tag{48}$$

Using CB, RB and the position of their origins, (fbx, fby) and (iox_1, ioy_1), we can form the bound-shape-function, BSF, of their closest-approach shape-boundary as:

$$BSF = BoundShapeFunction(ClosestApproachBoundary(RB, CB), \\ (DiffVar)fbx, (DiffVar)fby, (DiffVar)iox, (DiffVar)ioy) \tag{49}$$

This allows us to formulate the abutment-constraint between FB and the relaxed-boundary of IO_1 as:

$$BSF.evaluteFunct() = 1 \tag{50}$$

2.2.5 Initial Layout

To solve final-floorplanning optimization-problem, we need a set of initial values for its optimization variables. Specifically, we need initial values for:

- Physical-boundary origins, widths, and heights
- Net bounding-box origins, widths, and heights

- Floorplan-boundary origin, width, and height.
- I/O-block origins

While we can generate these values directly from the simplified-floorplan, we would likely start with a floorplan-boundary, the bounding box enclosing the macro-block physical-boundaries of the simplified-floorplan, that is significantly larger than necessary as the simplified-floorplan was generated without regard to floorplan area. Consequently, we have an opportunity to compact this floorplan while maintaining its total half-perimeter wire-length. This would give us a better set of initial values with which to initialize the final-floorplanning problem. This leads to an intermediate floorplan-compaction problem.

A formal description of this Fixed-pin Single-boundary Simplified-floorplan Compaction problem can be stated as follows:

Given:

- An axis aligned rectangle which is a bounding-box representing the floorplan-boundary, FB, with its origin, (fbx, fby), located at the center of the rectangle, and having a height fbh, and width fbw.
- A set, M, containing m macro-block physical-boundaries where each macro-block physical-boundary, i, is an axis-aligned rectangle whose origin, (mx_i, my_i), located at the center of the rectangle and has a height mh_i, width mw_i, and a fixed area ma_i.
- A set, N, containing n net bounding-boxes where each of the net bounding-boxes, j, is an axis-aligned rectangle with its origin, (nx_j, ny_j), located at the center of the rectangle and has a height nh_j, and width nw_j.
- Associated with each of the n net bounding-boxes in N, a set of macro-block pins, P_j, containing p_j pins where each of the pins, k, is located at $(px_{j,k}, py_{j,k})$ which, for the fixed-pin case, is the origin of its corresponding macro-block, (mx_i, my_i).
- The total half-perimeter wire-length, $STWL$, of the corresponding simplified-floorplan.

Determine:

- The location of the origin, (fbx, fby), the height, fbh, and the width, fbw, of the floorplan-boundary FB.
- The location of the origin, (mx_i, my_i), the height, mh_i, and the width, mw_i, for each of the m physical-boundaries in M.
- The location of the origin, (nx_j, ny_j), the height, nh_j, and the width, nw_j, for each of the n net bounding-boxes in N.

Such that:

- The perimeter of the floorplan-boundary bounding-box, FB, is minimized.
- The total half-perimeter wire-length of the n net bounding-boxes, in N, does not exceed a fixed upper bound greater than or equal to $STWL$.
- No two physical-boundaries in M overlap.
- The floorplan-boundary axis-aligned bounding-box, FB, encloses all of the m physical-boundaries in M.

- The aspect ratio for each of the m physical-boundaries in M is between 0.5 and 2.
- The area of each of the m physical-boundaries, i, in M is ma_i
- For each of the n nets, j, each pin, k, in P_j is contained in the axis-aligned bounding-box for net j.

From this, we can formulate the Fixed-pin Single-boundary Compaction problem as the following high-level optimization:

Minimize Floorplan-boundary Perimeter

while satisfying the following six sets of constraints:

1. Nonoverlap constraints between all pairs of physical-boundaries in M.
2. Physical-boundary aspect ratio constraints.
3. Physical-boundary area constraints.
4. Containment constraints to keep each fixed-pin contained in its associated net's bounding-box.
5. Containment constraints to keep each physical-boundary contained in the floorplan-boundary.
6. Total Half-Perimeter Wire-length upper bound constraint.

For the Nonoverlap constraints (constraint 1) we will continue to use the relaxed boundaries of *section 2.1.8* in order to take advantage of improved macro-block mobility. For the upper bound on the total half-perimeter wire-length (constraint 6) we could, under ideal circumstances, set the upper bound to $STWL$. However, due to round-off errors, this approach can lead to optimization convergence problems. To avoid these convergence problems and to provide a larger degree of compaction, we set the upper bound to be 102% of $STWL$, giving us an upper bound constraint of:

$$THPWL \leq 1.02 \cdot STWL \tag{51}$$

To solve this compaction optimization-problem, we need a set of initial values for its optimization variables. That is, we need initial values for:

- Physical-boundary origins, widths, and heights.
- Net bounding-box origins, widths, and heights.
- Floorplan-boundary origin, width, and height.

The macro-block origins, widths, and heights as well as the net bounding-box origins, width and height are taken directly form the simplified-floorplan with relaxed boundaries (*section 2.1.9, Figure 11*). The floorplan-boundary origin, width and height are taken to be the origin, width and height of the bounding-box enclosing the macro-block physical-boundaries of the same simplified-floorplan.

The resulting compacted simplified-floorplan, shown in *Figure 15*, has a total half-perimeter wire-length of 90.2, which, as expected, is a 2% increase in total half-perimeter wire-length over the corresponding simplified-floorplan (*Figure 11*) while providing a significant reduction in floorplan-boundary area.

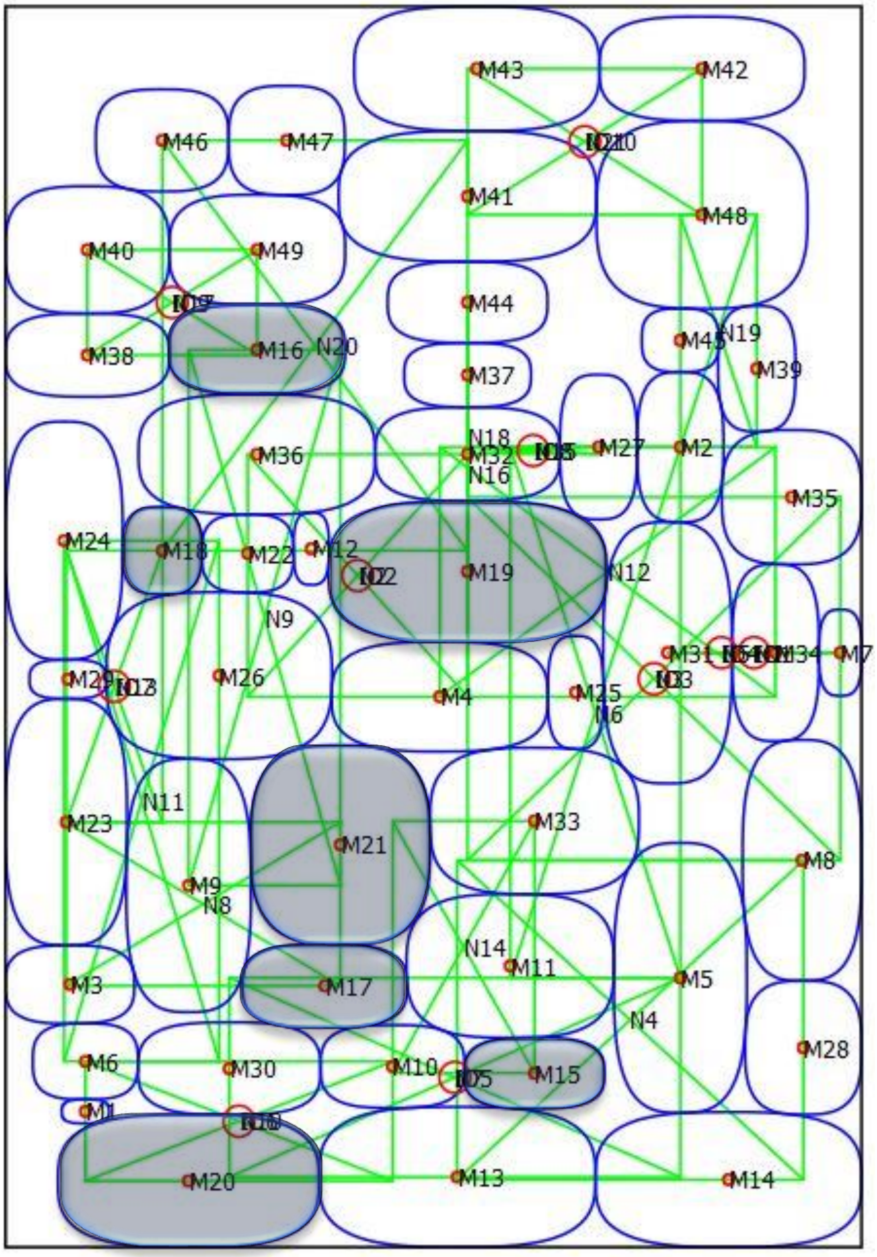

Figure 15: Fixed-pin Single-boundary Compacted Simplified-floorplan with 10 relaxed I/O-block boundaries.

2.2.6 Final Floorplanning with Relaxed Boundaries

With this compacted simplified-floorplan we can now generate the set of initial values needed to solve final-floorplanning optimization-problem. Specifically, the macro-block origins, widths, and heights as well as the net bounding-box origins, widths and heights are taken directly form the compacted simplified-floorplan. Additionally, the floorplan-boundary origin, width, and height are set to the origin, width, and height of the bounding-box enclosing the macro-block relaxed physical-boundaries of the same compacted simplified-floorplan. The origin for each of the I/O-blocks, as shown in

Figure 15, are set to the origin of their corresponding net bounding-box in the compacted simplified-floorplan.

Solving the Fixed-pin Single-boundary Final-floorplanning problem with relaxed boundaries produces the floorplan shown in *Figure 16*. The resulting floorplan has a total half-perimeter wire-length of 99.1 which is a 12.1% increase over the simplified-floorplan with relaxed macro-block boundaries shown in *Figure 11*.

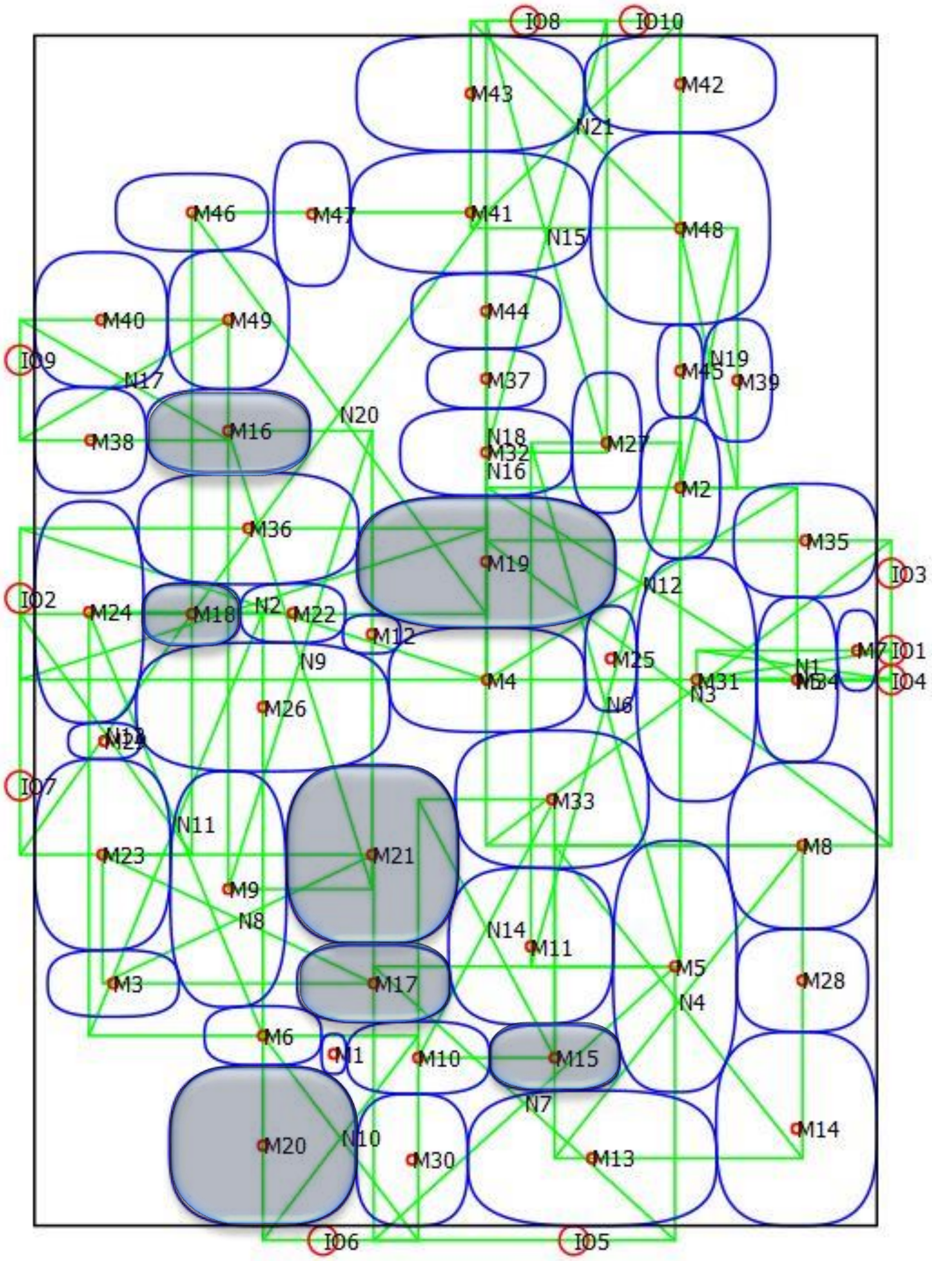

Figure 16: Fixed-pin Single-boundary Final-floorplan with relaxed boundaries.

2.2.7 Legalized Final-Floorplan

As was the case in *section 2.1.9*, floorplanning with relaxed shapes can result in unresolved overlaps between the actual rectangular-shapes. To resolve these overlaps and legalize the floorplan we will, again, re-solve the final-floorplanning problem using macro-block nonoverlap constraints constructed from the macro-block rectangular boundaries and the I/O-block nonoverlap and abutment constraints constructed using the I/O-block square boundaries. Further, we initialize the optimization variables to the values produced by the final-floorplan with relaxed boundaries. Finally, we solve this floorplan legalization problem using IPOPT's warm-start capability in order to ensure that the resulting legalized floorplan does not stray too far from the floorplan produced using the relaxed boundaries.

The resulting legalized floorplan, shown in *Figure 17*, has a total half-perimeter wire-length of 101.0. This is a 12.1% increase over the simplified-floorplan with rectangular macro-block boundaries shown in *Figure 13*.

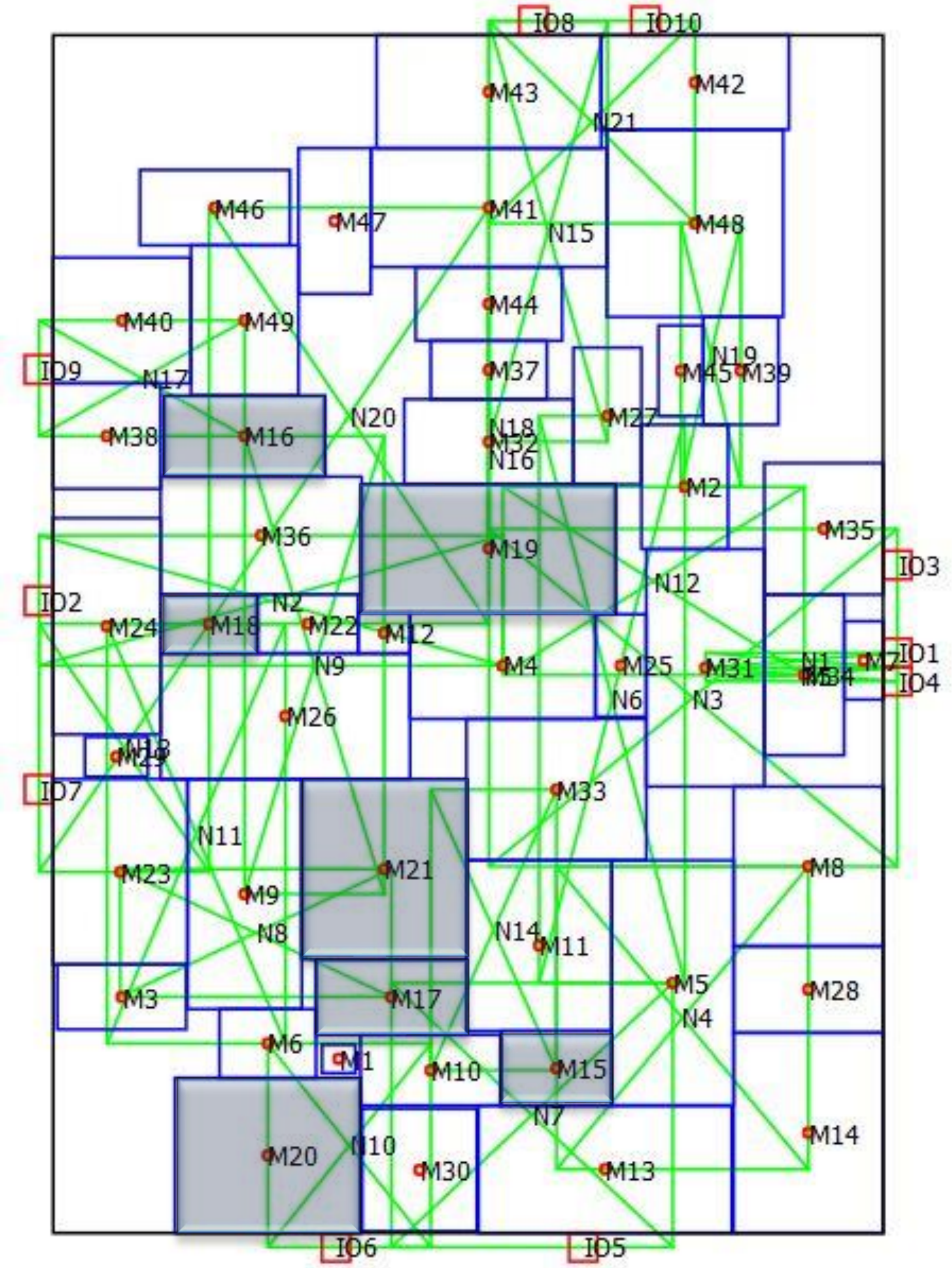

Figure 17: Fixed-pin Single-boundary Final-floorplan with rectangular boundaries.

3 Dynamic-Pin Single-Boundary Floorplanning

In this section we will build on the work of *Section 2* and develop a solution to the Dynamic-pin Single-boundary Floorplanning problem. Again, we will determine the location and aspect ratio of a set of soft macro-blocks as well as the location of a set of fixed-size I/O-blocks and the dimensions of the floorplan boundary. However, in addition, we will also need to determine the location for each macro-block's dynamic-pin.

We will, again, solve this floorplanning problem using the same two-stage successive-refinement strategy. The first, simplified-floorplanning, stage will solve the floorplanning problem in the absence of floorplan-boundary and I/O-block information. The second, Final-floorplanning stage, will again generate the complete floorplan by refining the simplified-floorplan using an analytical optimization which now includes the floorplan-boundary and I/O-block information. We will initialize this stage using the macro-block position and aspect ratios of the simplified-floorplan and derive the initial floorplan-boundary shape and initial position for each of the I/O-blocks from information extracted from the simplified-floorplan.

The primary difference between the solution developed in this section over that developed in *Section 2* is the addition of a new set of optimization variables, representing the dynamic-pin locations, and an additional set of constraints to limit the movement of the dynamic-pins.

3.1 Simplified Floorplanning

A formal description of the Simplified Dynamic-pin Single-boundary Floorplanning problem can be stated as follows:

Given:

- A set, M, containing m macro-block physical-boundaries where each macro-block physical-boundary, i, is an axis-aligned rectangle whose origin, (mx_i, my_i), located at the center of the rectangle and has a height mh_i, width mw_i, and a fixed area ma_i.
- A set, DP, containing m macro-block dynamic-pin positions where each dynamic-pin position, i, is a point, (dpx_i, dpy_i).
- A set, N, containing n net bounding-boxes where each of the net bounding-boxes, j, is an axis-aligned rectangle with its origin, (nx_j, ny_j), located at the center of the rectangle and has a height nh_j, and width nw_j.
- Associated with each of the n net bounding-boxes in N, a set of macro-block pins, P_j, containing p_j pins where each of the pins, k, is located at $(px_{j,k}, py_{j,k})$ which, for the dynamic-pin case, is the corresponding macro-block dynamic-pin position, (dpx_i, dpy_i).

Determine:

- The location of the origin, (mx_i, my_i), the height, mh_i, and the width, mw_i, for each of the m physical-boundaries in M.
- The location of the origin, (nx_j, ny_j), the height, nh_j, and the width, nw_j, for each of the n net bounding-boxes in N.
- The location of the dynamic-pin, $(dpx_{i,}, dpy_{i,})$, for each of the m dynamic-pins in DP.

Such that:

- The total half-perimeter wire-length of the n net bounding-boxes, in N, is minimized.
- No two physical-boundaries in M overlap.
- The aspect ratio for each of the m physical-boundaries is between 0.5 and 2.
- The area of each of the m physical boundaries is ma_i

- For each of the n nets, j, each pin, k, in P_j is contained in net j's bounding-box.
- Each of the m dynamic-pin positions, i, in DP, is contained in its dynamic-pin containment-boundary defined as an axis-aligned rectangle whose origin, (mx_i, my_i), is located at the center of the rectangle with a height $0.5 \cdot mh_i$ and width $0.5 \cdot mw_i$.

From this, we can formulate the Simplified Dynamic-pin Single-boundary Floorplanning problem as the following high-level optimization:

Minimize Total Half-Perimeter Wire-length

while satisfying the following five sets of constraints:

1. Nonoverlap constraints between all pairs of physical-boundaries in M.
2. Physical-boundary aspect ratio constraints.
3. Physical-boundary area constraints.
4. Containment constraints to keep each dynamic-pin contained in the bounding-box of each of its associated nets.
5. Containment constraints to keep each dynamic-pin contained in its dynamic-pin containment-boundary.

3.1.1 Dynamic-Pin Containment Constraints

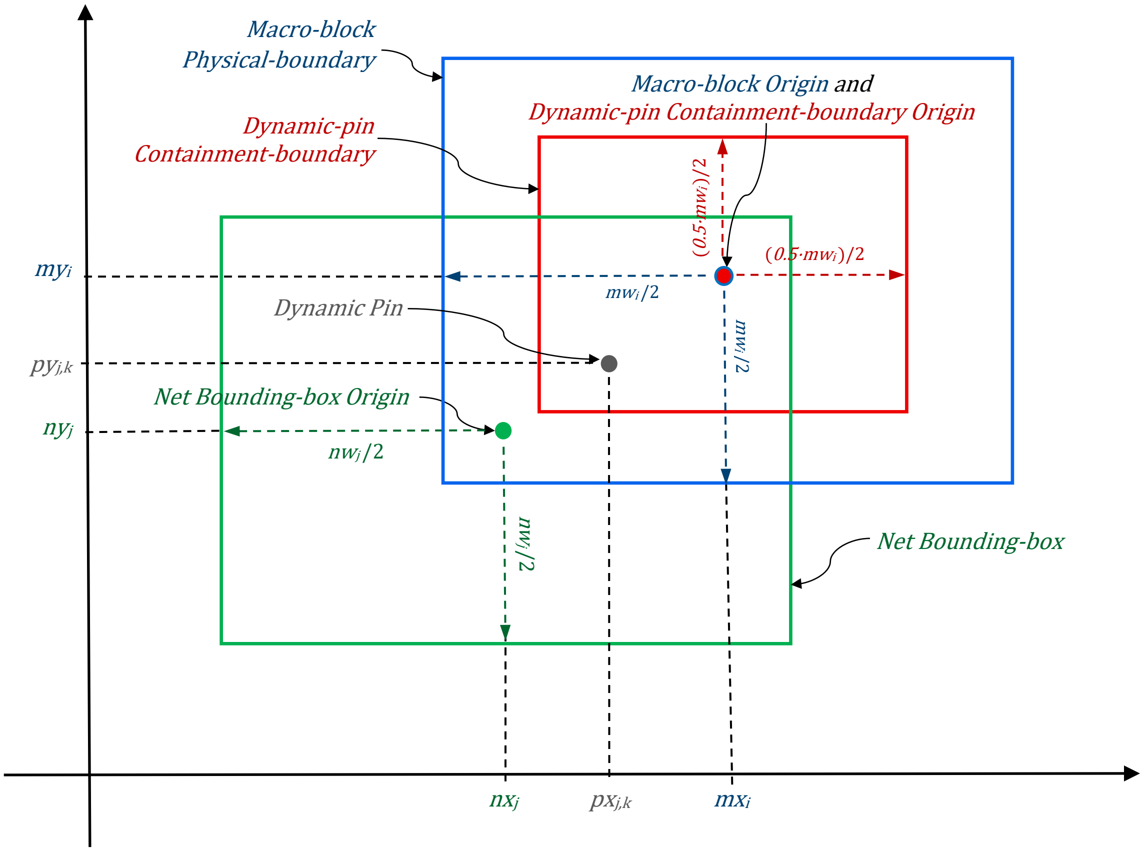

Figure 18: Spatial relationships for dynamic-pin containment constraints.

To formulate the dynamic-pin containment constraints (constraints 4 and 5), we begin by reformulating the net-containment constraints (constraint 4), developed in *Section 2.1.4*, to account for the mobility of the dynamic pins. Specifically, as illustrated by the spatial relationships shown in *Figure 18*, this set of constraints is reformulated as:

$$px_{j,k} \leq nx_j + nw_j/2 \tag{52}$$

$$px_{j,k} \geq nx_j - nw_j/2 \tag{53}$$

$$py_{j,k} \leq ny_j + nh_j/2 \tag{54}$$

$$py_{j,k} \geq ny_j - nh_j/2 \tag{55}$$

Similarly, the constraints that make sure that the dynamic pins stay within their macro-block's dynamic-pin containment boundary (constraint 5) are, as illustrated in *Figure 18*, formulated as:

$$px_{j,k} \leq mx_i + mw_i/4 \tag{56}$$

$$px_{j,k} \geq mx_i - mw_i/4 \tag{57}$$

$$py_{j,k} \leq my_i + mh_i/4 \tag{58}$$

$$py_{j,k} \geq my_i - mh_i/4 \tag{59}$$

3.1.2 Simplified Floorplanning with Relaxed Boundaries

To solve Simplified-floorplanning optimization-problem, we need a set of initial values for its optimization variables. Specifically, we need initial values for:

- Physical-boundary origins, widths, and heights
- Net bounding-box origins, widths, and heights
- Dynamic-pin positions

Starting from a ***10X*** initial spacing, we can select the origin (mx_i, my_i), widith (mw_i), and height (mh_i) for each of the physical-boundaries as outlined in Section 2.1.5. The dynamic-pin positions (dpx_i, dpy_i) are set to the origin of their corresponding macro-block (mx_i, my_i). From this, each net's origin $(nx_j,\ ny_j)$, width (nw_j), and height (nh_j) are selected such that they form the minimum sized bounding-box enclosing all of that net's dynamic-pins.

Using these initial values to solve the Dynamic-pin Single-boundary Simplified-floorplanning problem with relaxed boundaries produces the floorplan shown in *Figure 19*. The total half-perimeter wire-length of this floorplan is 72.1 which is a 18.4% improvement over that of the fix-pin floorplan shown in *Figure 11*.

Note that, as shown in *Figure 19*, the location of a dynamic-pin can take on any value in the set defined by the intersection of the dynamic-pin's containment-boundary and each of the net bounding-boxes that contain the pin. To illustrate this idea, consider the dynamic-pins for macro-blocks 8, 11, and 47 shown in *Figure 19*:

- The location of the dynamic-pin for macro-block 8 can take on only one value, the point defined by the intersection of its dynamic-pin's containment boundary and the bounding boxes for nets 3 and 4.

- The dynamic-pin for Macro-block 11 can take on any value from the line segment defined by the intersection of its dynamic-pin’s containment-boundary and the bounding boxes for nets 6 and 14.

- The dynamic-pin for Macro-block 47, as placed, can take on any value from the rectangle defined by the intersection of its dynamic-pin’s containment boundary and the bounding box for net 20.

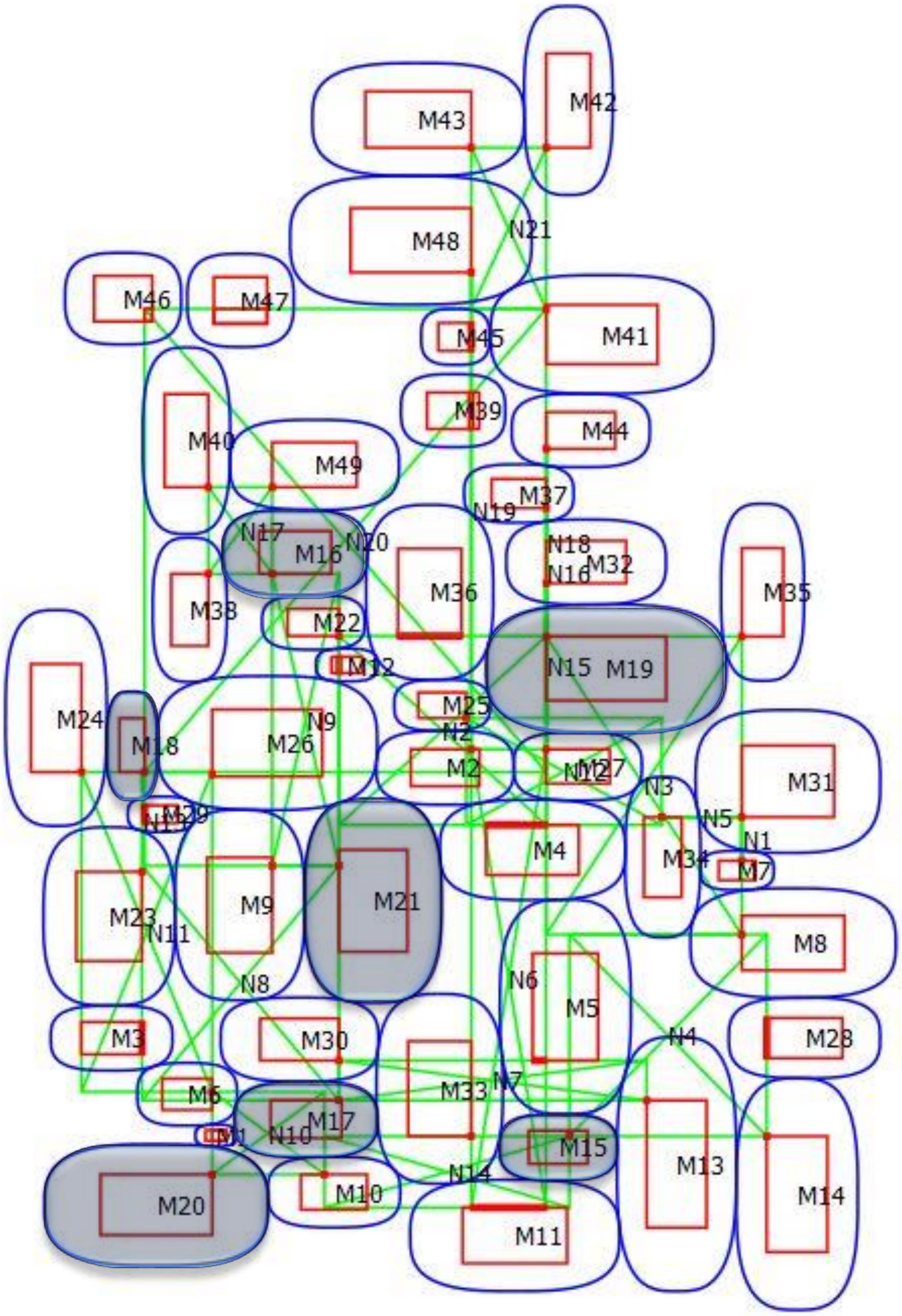

Figure 19: Dynamic-pin Single-boundary Simplified-floorplan starting from *10X* initial spacing and relaxed macro-block physical-boundaries.

3.2 Final Floorplanning

In the final-floorplanning stage we will use the information derived from the relaxed simplified-floorplan to provide the starting point for solving the full Dynamic-pin Single-Boundary Floorplanning problem which can be stated as follows:

Given:

- An axis-aligned rectangle which is a bounding-box representing the floorplan-boundary, FB, with its origin, (fbx, fby), located at the center of the rectangle, and having a height fbh, and width fbw.
- A set, M, containing m macro-block physical-boundaries where each macro-block physical-boundary, i, is an axis-aligned rectangle whose origin, (mx_i, my_i), located at the center of the rectangle and has a height mh_i, width mw_i, and a fixed area ma_i.
- A set, DP, containing m macro-block dynamic-pin positions where each dynamic-pin position, i, is a point, (dpx_i, dpy_i).
- A set, N, containing n net bounding-boxes where each of the net bounding-boxes, j, is an axis-aligned rectangle with its origin, (nx_j, ny_j), located at the center of the rectangle and has a height nh_j, and width nw_j.
- Associated with each of the n net bounding-boxes in N, a set of macro-block pins, P_j, containing p_j pins where each of the pins, k, is located at $(px_{j,k}, py_{j,k})$ which, for the dynamic-pin case, is the corresponding macro-block dynamic-pin position, (dpx_i, dpy_i).
- A set, IO, containing io I/O-block boundaries where each I/O-block boundary, q, is an axis-aligned square whose origin, (iox_q, ioy_q), located at the center of the square and all I/O-block boundaries have the same fixed height and width, $iohw$. Each of the io I/O-blocks in IO is associated with one of the n nets in N, ion_q, and no net can be associated with more than one I/O-block ($io \leq n$).

Determine:

- The location of the origin, (fbx, fby), the height, fbh, and the width, fbw, of the floorplan-boundary FB.
- The location of the origin, (mx_i, my_i), the height, mh_i, and the width, mw_i, for each of the m physical-boundaries in M.
- The location of the origin, (nx_j, ny_j), the height, nh_j, and the width, nw_j, for each of the n net bounding-boxes in N.
- The location of the origin, (iox_q, ioy_q), for each of the io fixed-size I/O-blocks in IO.
- The location of the dynamic-pin, $(dpx_{i,}, dpy_{i,})$, for each of the m dynamic-pins in DP.

Such that:

- The total half-perimeter wire-length of the n net bounding-boxes, in N, is minimized.
- No two physical-boundaries in M overlap.
- The floorplan-boundary bounding-box, FB, encloses all of the m physical-boundaries in M.

- The aspect ratio for each of the m physical-boundaries in M is between 0.5 and 2.
- The area of each of the m physical-boundaries, i, in M is ma_i
- For each of the n nets, j, each pin, k, in P_j is contained in the bounding-box for net j.
- Each of the io I/O-blocks, q, in IO has its origin, (iox_q, ioy_q), contained in the bounding box for net ion_q.
- No two of the io I/O-block boundaries in IO overlap.
- Each of the io I/O-block boundaries abuts the floorplan-boundary bounding-box, FB.
- Each of the m dynamic-pin positions, i, in DP, is contained in its dynamic-pin containment-boundary defined as an axis-aligned rectangle whose origin, (mx_i, my_i), is located at the center of the rectangle with a height $0.5 \cdot mh_i$ and width $0.5 \cdot mw_i$.

From this, we can formulate the Final Dynamic-pin Single-boundary Floorplanning problem as the following high-level optimization:

Minimize Total Half-Perimeter Wire-length

while satisfying the following nine sets of constraints:

1. Nonoverlap constraints between all pairs of physical-boundaries in M.
2. Physical-boundary aspect ratio constraints.
3. Physical-boundary area constraints.
4. Containment constraints to keep each dynamic-pin contained in its associated net's bounding-box.
5. Containment constraints to keep each I/O-pin contained in it associated net's bounding-box.
6. Containment constraints to keep each physical-boundary contained in floorplan-boundary constraints
7. Abutment constraints to abut each I/O-block to the floorplan-boundary.
8. Nonoverlap constrains between all pairs of I/O-block boundaries in IO.
9. Containment constraints to keep each dynamic-pin contained in its dynamic-pin containment-boundary.

3.2.1 Initial Layout

To solve final-floorplanning optimization-problem, we need a set of initial values for its optimization variables. Specifically, we need initial values for:

- Physical-boundary origins, widths, and heights
- Net bounding-box origins, widths, and heights
- Floorplan-boundary origin, width, and height.
- I/O-block origins
- Dynamic-pin positions

Again, while we can generate these values directly from the simplified-floorplan, we would likely start with a floorplan-boundary that is significantly larger than necessary. Again, we can compact this floorplan while maintaining its total half-perimeter wire-length which will give us a better set of initial values with which to initialize the final-floorplanning problem.

A formal description of this Dynamic-pin Single-boundary Simplified-floorplan Compaction problem can be stated as follows:

Given:

- An axis aligned rectangle which is a bounding-box representing the floorplan-boundary, FB, with its origin, (fbx, fby), located at the center of the rectangle, and having a height fbh, and width fbw.
- A set, M, containing m macro-block physical-boundaries where each macro-block physical-boundary, i, is an axis-aligned rectangle whose origin, (mx_i, my_i), located at the center of the rectangle and has a height mh_i, width mw_i, and a fixed area ma_i.
- A set, DP, containing m macro-block dynamic-pin positions where each dynamic-pin position, i, is a point, (dpx_i, dpy_i).
- A set, N, containing n net bounding-boxes where each of the net bounding-boxes, j, is an axis-aligned rectangle with its origin, (nx_j, ny_j), located at the center of the rectangle and has a height nh_j, and width nw_j.
- Associated with each of the n net bounding-boxes in N, a set of macro-block pins, P_j, containing p_j pins where each of the pins, k, is located at $(px_{j,k}, py_{j,k})$ which, for the fixed-pin case, is the origin of its corresponding macro-block, (mx_i, my_i).
- The total half-perimeter wire-length, $STWL$, of the corresponding simplified-floorplan.

Determine:

- The location of the origin, (fbx, fby), the height, fbh, and the width, fbw, of the floorplan-boundary FB.
- The location of the origin, (mx_i, my_i), the height, mh_i, and the width, mw_i, for each of the m physical-boundaries in M.
- The location of the origin, (nx_j, ny_j), the height, nh_j, and the width, nw_j, for each of the n net bounding-boxes in N.
- The location of the dynamic-pin, $(dpx_{i,}, dpy_{i,})$, for each of the m dynamic-pins in DP.

Such that:

- The perimeter of the floorplan-boundary bounding-box, FB, is minimized.
- The total half-perimeter wire-length of the n net bounding-boxes, in N, does not exceed a fixed upper bound greater than or equal to $STWL$.
- No two physical-boundaries in M overlap.
- The floorplan-boundary, FB, encloses all of the physical-boundaries in M.
- The aspect ratio for each of the physical-boundaries in M is between 0.5 and 2.
- The area of each of the physical-boundaries, i, in M is ma_i
- For each of the n nets, j, each pin, k, in P_j is contained in the bounding-box for net j.
- Each of the m dynamic-pin positions, i, in DP, is contained in its dynamic-pin containment-boundary defined as an axis-aligned rectangle

whose origin, (mx_i, my_i), is located at the center of the rectangle with a height $0.5 \cdot mh_i$ and width $0.5 \cdot mw_i$.

From this, we can formulate the Dynamic-pin Single-boundary Compaction problem as the following high-level optimization:

Minimize Floorplan-boundary Perimeter

while satisfying the following seven sets of constraints:

1. Nonoverlap constraints between all pairs of physical-boundaries in *M*.
2. Physical-boundary aspect ratio constraints.
3. Physical-boundary area constraints.
4. Containment constraints to keep each dynamic-pin contained in its associated net's bounding-box.
5. Containment constraints to keep each physical-boundary contained in the floorplan-boundary.
6. Total Half-Perimeter Wire-length upper bound constraint.
7. Containment constraints to keep each dynamic-pin contained in its dynamic-pin containment-boundary.

For the Nonoverlap constraints (constraint 1) we will continue to use the relaxed boundaries of *section 2.1.8* in order to take advantage of improved macro-block mobility. For the upper bound on the total half-perimeter wire-length (constraint 6) we will, again, set the upper bound to be 102% of *STWL*, giving us an upper bound constraint of:

$$THPWL \leq 1.02 \cdot STWL \qquad (60)$$

To solve this compaction optimization-problem, we need a set of initial values for its optimization variables. That is, we need initial values for:

- Physical-boundary origins, widths, and heights.
- Net bounding-box origins, widths, and heights.
- Floorplan-boundary origin, width, and height.
- Dynamic-pin positions

The macro-block origins, widths, and heights as well as the net bounding-box origins, width, and height and the dynamic-pin positions are taken directly form the simplified-floorplan with relaxed boundaries (*Section 3.1.2, Figure 19*). The floorplan-boundary origin, width and height are taken to be the origin, width and height of the bounding-box enclosing the macro-block physical-boundaries of the same simplified-floorplan.

The resulting compacted simplified-floorplan, shown in *Figure 20*, has a total half-perimeter wire-length of 73.6, which, as expected, is a 2% increase in total half-perimeter wire-length over the corresponding simplified-floorplan (*Figure 19*) while providing a significant reduction in floorplan-boundary area.

3.2.2 Final Floorplan with Relaxed Boundaries

Again, as outlined in *Section 2.2.6*, this compacted simplified-floorplan is used to generate the set of initial values needed to solve final-floorplanning optimization-problem. Specifically, the macro-block origins, widths, and heights as well as the dynamic-pin locations and net bounding-box origins, widths and heights are taken directly form the compacted simplified-floorplan. Additionally, the floorplan-boundary origin, width, and height are set to the origin, width, and height of the bounding-box enclosing the macro-block relaxed physical-boundaries of the same compacted simplified-floorplan. The origin for each of the I/O-blocks, as shown in *Figure 20*, are set to the origin of their corresponding net bounding-box in the compacted simplified-floorplan.

The resulting Dynamic-pin Single-boundary Final-floorplan with relaxed boundaries, shown in *Figure 21*, has a total half-perimeter wire-length of 88.0. This is an 11% reduction over the fixed-pin floorplan of *Section 2.2.6* shown in *Figure 16*.

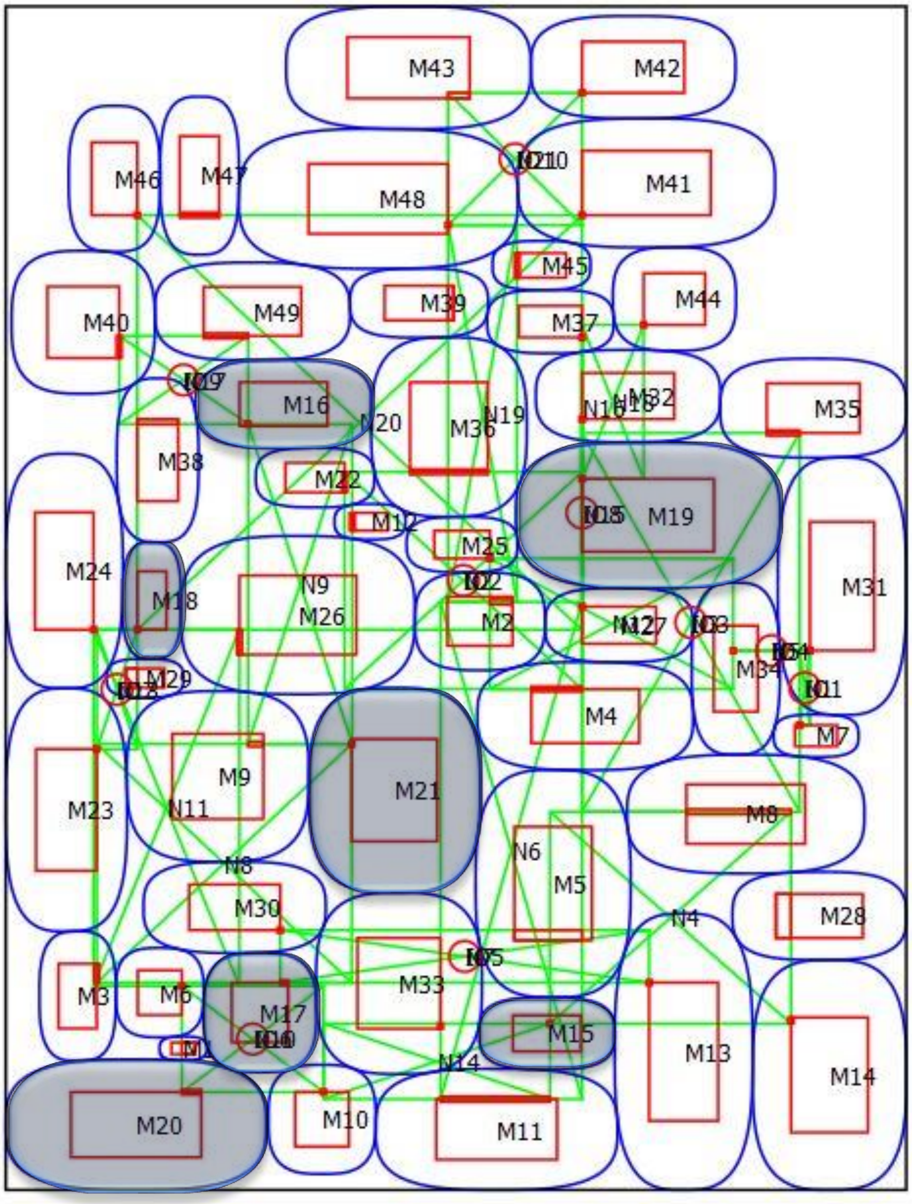

Figure 20: Dynamic-pin Single-boundary Compacted Simplified-floorplan with 10 relaxed I/O-block boundaries.

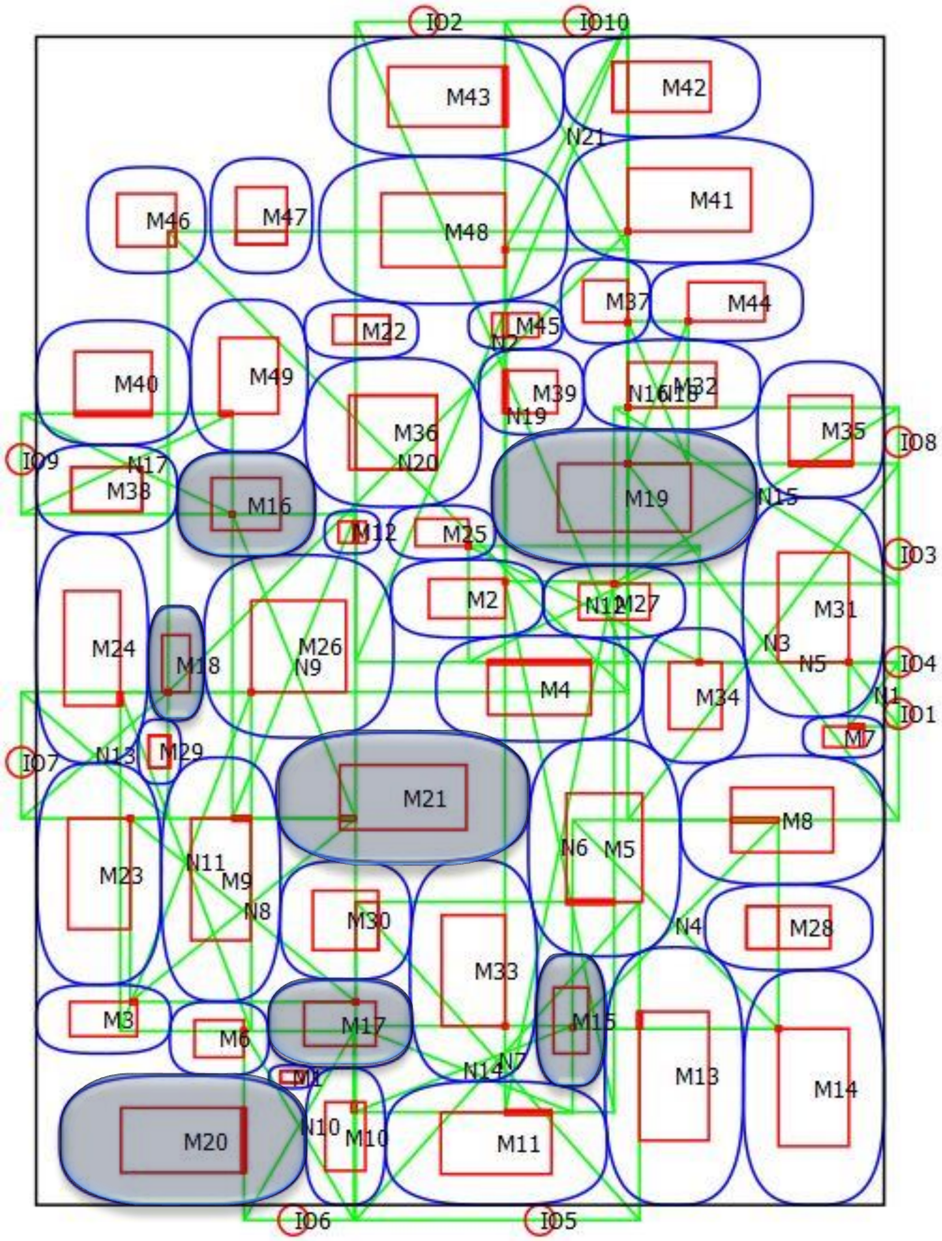

Figure 21: Dynamic-pin Single-boundary Final-floorplan with relaxed boundaries.

3.2.3 Legalized Final-Floorplan

Again, floorplanning with relaxed shapes will result in unresolved overlaps between the actual rectangular-shapes. To resolve these overlaps and legalize the floorplan we will, again, re-solve the final-floorplanning problem using macro-block nonoverlap constraints constructed from the macro-block rectangular boundaries and the I/O-block nonoverlap and abutment constraints constructed using the I/O-block square boundaries. Further, we initialize the optimization variables to the values produced by the final-floorplan with relaxed boundaries. Finally, we solve this floorplan legalization problem using IPOPT's warm-start capability in order to ensure that the resulting legalized floorplan does not stray too far from the floorplan produced using the relaxed boundaries.

The resulting legalized floorplan, shown in *Figure 22*, has a total half-perimeter wire-length of 89.4. This is an 11.5% reduction over the fixed-pin floorplan of *Section 2.2.7* shown in *Figure 17*.

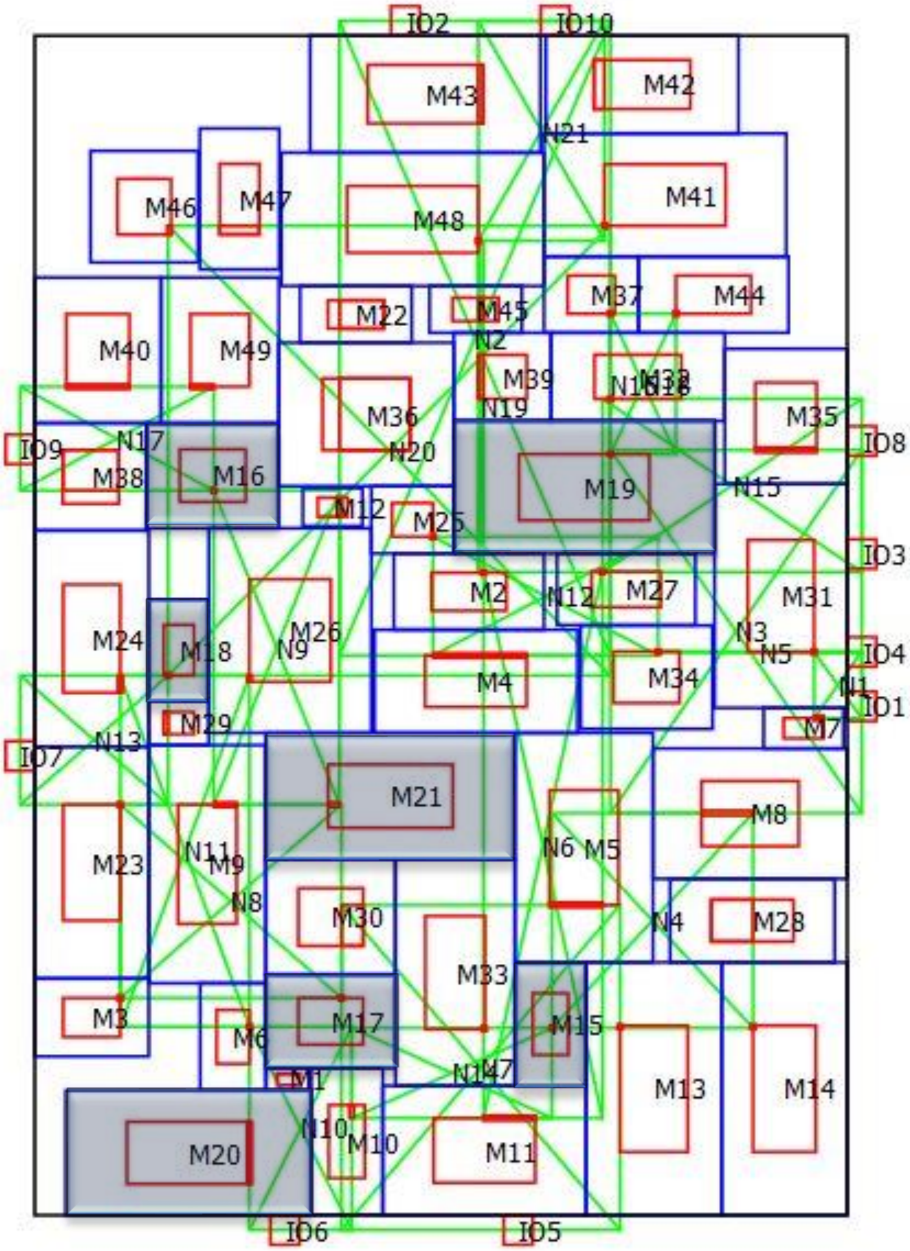

Figure 22: Dynamic-pin Single-boundary Final-floorplan with rectangular boundaries.

4 Dynamic-Pin Multi-Boundary Floorplanning

In this section we will build on the work of *Section 3* and develop a solution to the Dynamic-pin Multi-boundary Floorplanning problem. Using the same two-stage successive-refinement strategy developed in *sections 2* and *3,* we will determine the location and aspect ratio of a set of soft macro-blocks as well as the location of a set of fixed-size I/O-blocks, the dimensions of the floorplan boundary and the location for each macro-block's dynamic-pin.

The primary difference between the solution developed in this section over that developed in *Section 3* is the addition of a set of constraints which prevent overlaps between a second set of macro-block boundaries, the macro-block's thermal-boundaries (as illustrated in *Section 1, Figure 3*).

4.1 Simplified Floorplanning

A formal description of the Simplified Dynamic-pin Multi-boundary Floorplanning problem can be stated as follows:

Given:

- A set, M, containing m macro-block physical-boundaries where each macro-block physical-boundary, i, is an axis-aligned rectangle whose origin, (mx_i, my_i), located at the center of the rectangle and has a height mh_i, width mw_i, and a fixed area ma_i.
- A set, TB, containing m macro-block thermal-boundaries where each macro-block thermal-boundary, i, is a cubic superellipse whose origin, (mx_i, my_i), is located at the center of the superellipse and has a height $sf_i \cdot mh_i$ and width $sf_i \cdot mw_i$ where sf_i is a fixed thermal scaling-factor for macro-block i.
- A set, DP, containing m macro-block dynamic-pin positions where each dynamic-pin position, i, is a point, (dpx_i, dpy_i).
- A set, N, containing n net bounding-boxes where each of the net bounding-boxes, j, is an axis-aligned rectangle with its origin, (nx_j, ny_j), located at the center of the rectangle and has a height nh_j, and width nw_j.
- Associated with each of the n net bounding-boxes in N, a set of macro-block pins, P_j, containing p_j pins where each of the pins, k, is located at $(px_{j,k}, py_{j,k})$ which, for the dynamic-pin case, is the corresponding macro-block dynamic-pin position, (dpx_i, dpy_i).

Determine:

- The location of the origin, (mx_i, my_i), the height, mh_i, and the width, mw_i, for each of the m physical-boundaries in M.
- The location of the origin, (nx_j, ny_j), the height, nh_j, and the width, nw_j, for each of the n net bounding-boxes in N.
- The location of the dynamic-pin, $(dpx_{i,}, dpy_{i,})$, for each of the m dynamic-pins in DP.

Such that:

- The total half-perimeter wire-length of the n net bounding-boxes, in N, is minimized.
- No two physical-boundaries in M overlap.
- The aspect ratio for each of the m physical-boundaries is between 0.5 and 2.
- The area of each of the m physical boundaries is ma_i
- For each of the n nets, j, each pin, k, in P_j is contained in net j's bounding-box.
- Each of the m dynamic-pin positions, i, in DP, is contained in its dynamic-pin containment-boundary defined as an axis-aligned rectangle whose origin, (mx_i, my_i), is located at the center of the rectangle with a height $0.5 \cdot mh_i$ and width $0.5 \cdot mw_i$.
- No two thermal-boundaries in TB overlap.

From this, we can formulate the Simplified Dynamic-pin Multi-boundary Floorplanning problem as the following high-level optimization:

Minimize Total Half-Perimeter Wire-length

while satisfying the following six sets of constraints:

1. Nonoverlap constraints between all pairs of physical-boundaries in M.
2. Physical-boundary aspect ratio constraints.
3. Physical-boundary area constraints.
4. Containment constraints to keep each dynamic-pin contained in the bounding-box of each of its associated nets.
5. Containment constraints to keep each dynamic-pin contained in the its dynamic-pin containment-boundary.
6. Nonoverlap constraints between all pairs of thermal-boundaries in TB.

4.1.1 Thermal-Boundary Nonoverlap Constraints

To construct the macro-block thermal-boundary nonoverlap constraints (constraint 6) we will again formulate these nonoverlap constraint using an ACGL bound-shape-function. That is, given two thermal-boundaries, TB_1 and TB_2 with thermal scaling-factors sf_1 and sf_2, from the set TB, we can form two scaled cubic-superellipse shape-boundaries, $SSEB_1$ and $SSEB_2$, as:

$$SSEB_1 = ScaledBoundary(SuperellipseBoundary(3.0,\ (DiffVar)mw_1,\ (DiffVar)mh_1),\ sf_1) \tag{61}$$

$$SSEB_2 = ScaledBoundary(SuperellipseBoundary(3.0,\ (DiffVar)mw_2,\ (DiffVar)mh_2),\ sf_2) \tag{62}$$

Using $SSEB_1$, $SSEB_2$ and the position of their origins, (mx_1, my_1) and (mx_2, my_2), we can form the bound-shape-function, BSF, of their closest-approach shape-boundary as:

$$BSF = BoundShapeFunction(ClosestApproachBoundary(SSEB_1, SSEB_2),\ (DiffVar)mx_1, (DiffVar)my_1, (DiffVar)mx_2, (DiffVar)my_2) \tag{63}$$

This allows us to formulate the nonoverlap-constraint between the thermal-boundaries of TB_1 and TB_2 as:

$$BSF.evaluteFunct() \geq 1 \tag{64}$$

and extract the gradient with respect to its differentiable variables mw_1, mh_1, mw_2, mh_2, mx_1, my_1, mx_2, and my_2 using:

$$GV = BSF.evaluateGrad() \tag{65}$$

where GV is the vector of partial derivatives for the bound-shape-function.

4.1.2 Simplified Floorplanning with Relaxed Boundaries

Solving the Dynamic-pin Multi-boundary Simplified-floorplanning problem with relaxed boundaries and a *10X* initial spacing produces the floorplan shown in *Figure 23*.

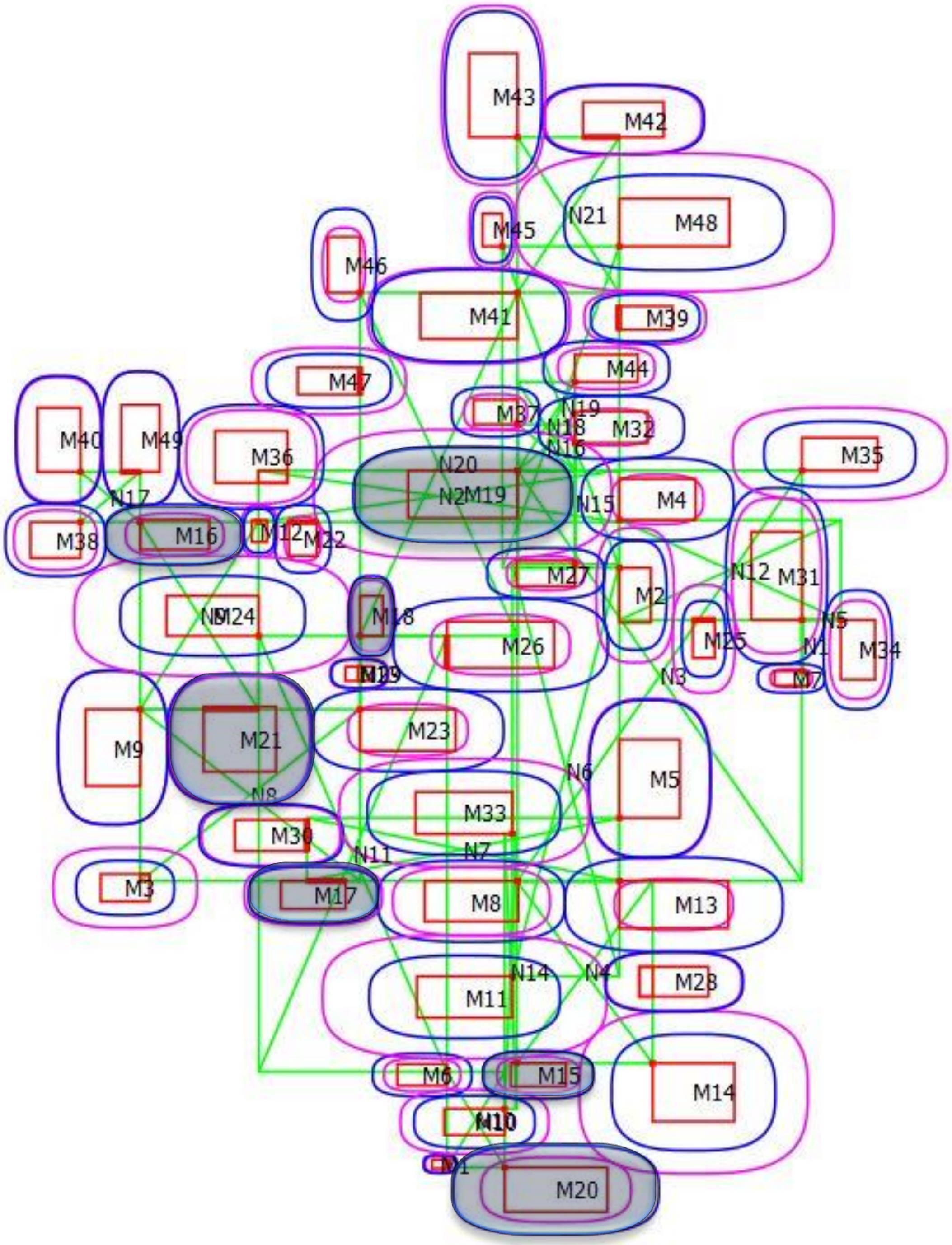

Figure 23: Dynamic-pin Multi-boundary Simplified-floorplan starting from *10X* initial spacing and relaxed macro-block physical-boundaries.

4.2 Final Floorplanning

In the final-floorplanning stage we will use the information derived from the relaxed simplified-floorplan to provide the starting point for solving the full Dynamic-pin Multi-Boundary Floorplanning problem. This Dynamic-pin Multi-Boundary Floorplanning problem can be stated as follows:

Given:

- An axis-aligned rectangle which is a bounding-box representing the floorplan-boundary, FB, with its origin, (fbx, fby), located at the center of the rectangle, and having a height fbh, and width fbw.
- A set, M, containing m macro-block physical-boundaries where each macro-block physical-boundary, i, is an axis-aligned rectangle whose origin, (mx_i, my_i), located at the center of the rectangle and has a height mh_i, width mw_i, and a fixed area ma_i.
- A set, TB, containing m macro-block thermal-boundaries where each macro-block thermal-boundary, i, is a cubic superellipse whose origin, (mx_i, my_i), is located at the center of the superellipse and has a height $sf_i \cdot mh_i$ and width $sf_i \cdot mw_i$ where sf_i is a fixed thermal scaling-factor for macro-block i.
- A set, DP, containing m macro-block dynamic-pin positions where each dynamic-pin position, i, is a point, (dpx_i, dpy_i).
- A set, N, containing n net bounding-boxes where each of the net bounding-boxes, j, is an axis-aligned rectangle with its origin, (nx_j, ny_j), located at the center of the rectangle and has a height nh_j, and width nw_j.
- Associated with each of the n net bounding-boxes in N, a set of macro-block pins, P_j, containing p_j pins where each of the pins, k, is located at $(px_{j,k}, py_{j,k})$ which, for the dynamic-pin case, is the corresponding macro-block dynamic-pin position, (dpx_i, dpy_i).
- A set, IO, containing io I/O-block boundaries where each I/O-block boundary, q, is an axis-aligned square whose origin, (iox_q, ioy_q), located at the center of the square and all I/O-block boundaries have the same fixed height and width, $iohw$. Each of the io I/O-blocks in IO is associated with one of the n nets in N, ion_q, and no net can be associated with more than one I/O-block ($io \leq n$).

Determine:

- The location of the origin, (fbx, fby), the height, fbh, and the width, fbw, of the floorplan-boundary FB.
- The location of the origin, (mx_i, my_i), the height, mh_i, and the width, mw_i, for each of the m physical-boundaries in M.
- The location of the origin, (nx_j, ny_j), the height, nh_j, and the width, nw_j, for each of the n net bounding-boxes in N.
- The location of the origin, (iox_q, ioy_q), for each of the io I/O-blocks in IO.
- The location of the dynamic-pin, $(dpx_{i,}, dpy_{i,})$, for each of the m dynamic-pins in DP.

Such that:

- The total half-perimeter wire-length of the n net bounding-boxes, in N, is minimized.
- No two physical-boundaries in M overlap.
- The floorplan-boundary, FB, encloses all of the m physical-boundaries in M.
- The aspect ratio for each of the m physical-boundaries in M is between 0.5 and 2.
- The area of each of the m physical-boundaries, i, in M is ma_i
- For each of the n nets, j, each pin, k, in P_j is contained in the bounding-box for net j.
- Each of the io I/O-blocks, q, in IO has its origin, (iox_q, ioy_q), contained in the bounding box for net ion_q.
- No two of the io I/O-block boundaries in IO overlap.
- Each of the io I/O-block boundaries abuts the floorplan-boundary, FB.
- Each of the m dynamic-pin positions, i, in DP, is contained in its dynamic-pin containment-boundary defined as an axis-aligned rectangle whose origin, (mx_i, my_i), is located at the center of the rectangle with a height $0.5 \cdot mh_i$ and width $0.5 \cdot mw_i$.
- No two thermal-boundaries in TB overlap.

From this, we can formulate the Final Dynamic-pin Multi-boundary Floorplanning problem as the following high-level optimization:

Minimize Total Half-Perimeter Wire-length

while satisfying the following ten sets of constraints:

1. Nonoverlap constraints between all pairs of physical-boundaries in M.
2. Physical-boundary aspect ratio constraints.
3. Physical-boundary area constraints.
4. Containment constraints to keep each dynamic-pin contained in its associated net's bounding-box.
5. Containment constraints to keep each I/O-pins contained in it associated net's bounding-box.
6. Containment constraints to keep each physical-boundary contained in the floorplan-boundary.
7. Abutment constraints to abut each I/O-block to the Floorplan-boundary.
8. Nonoverlap constrains between all pairs of I/O-block boundaries in IO.
9. Containment constraints to keep each dynamic-pin contained in its dynamic-pin containment-boundary.
10. Nonoverlap constraints between all pairs of thermal-boundaries in TB.

As was the case in *Sections 22.2.5* and *3.2.1* we can generate (left as an exercise for the reader) and solve a Dynamic-pin Multi-boundary Simplified-floorplan Compaction problem give us a better set of initial values with which to initialize the final-floorplanning problem.

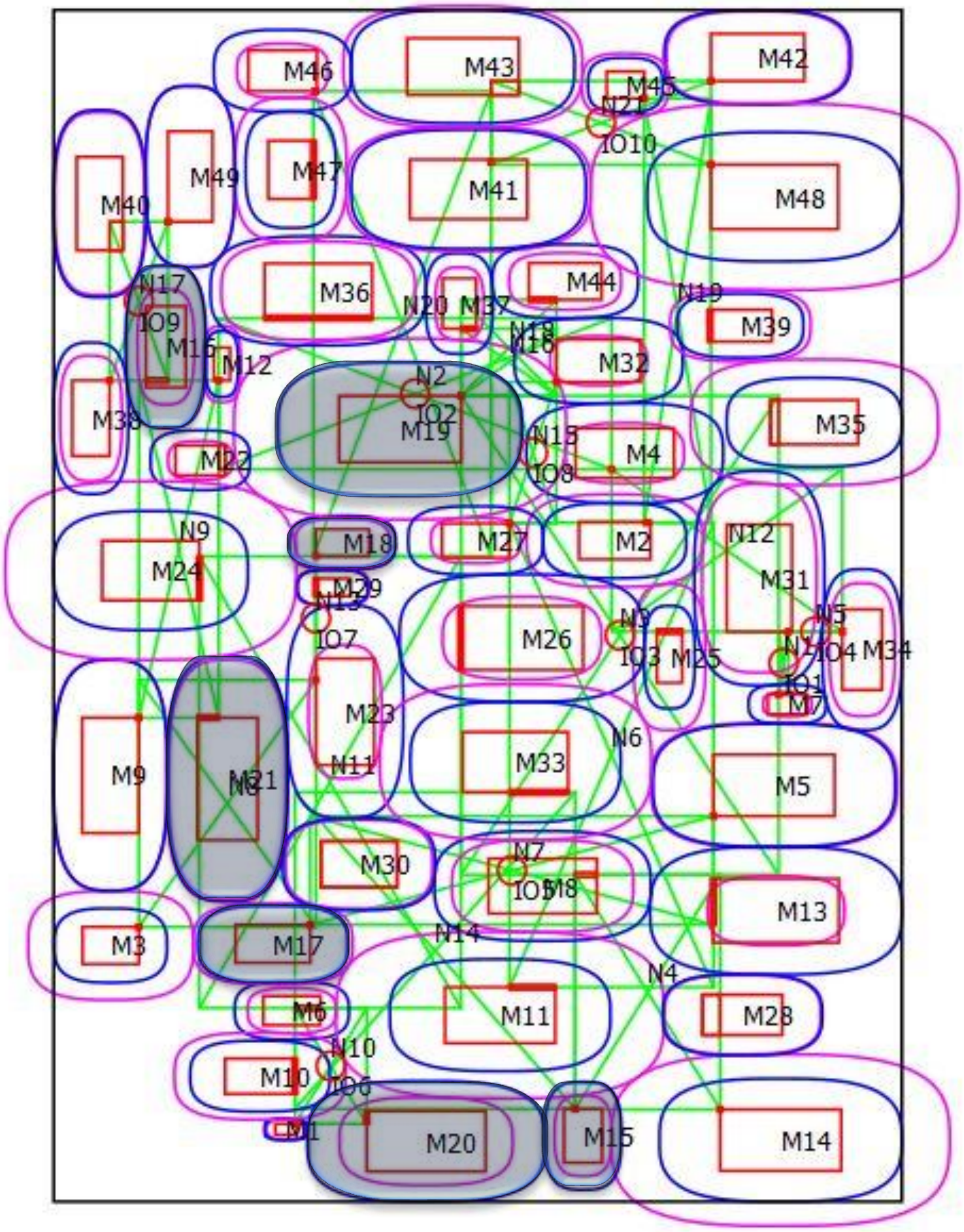

Figure 24: Dynamic-pin Multi-boundary Compacted Simplified-floorplan with 10 relaxed I/O-block boundaries.

The resulting compacted simplified-floorplan is shown in *Figure 24*. Note that, unlike the physical-boundaries, the thermal-boundaries are not constrained by the floorplan-boundary and thus are free to extend beyond the floorplan-boundary.

4.2.1 Final Floorplan with Relaxed Boundaries

Starting from this compacted floorplan, we can now generate the final-floorplan with relaxed boundaries shown in *Figure 25*.

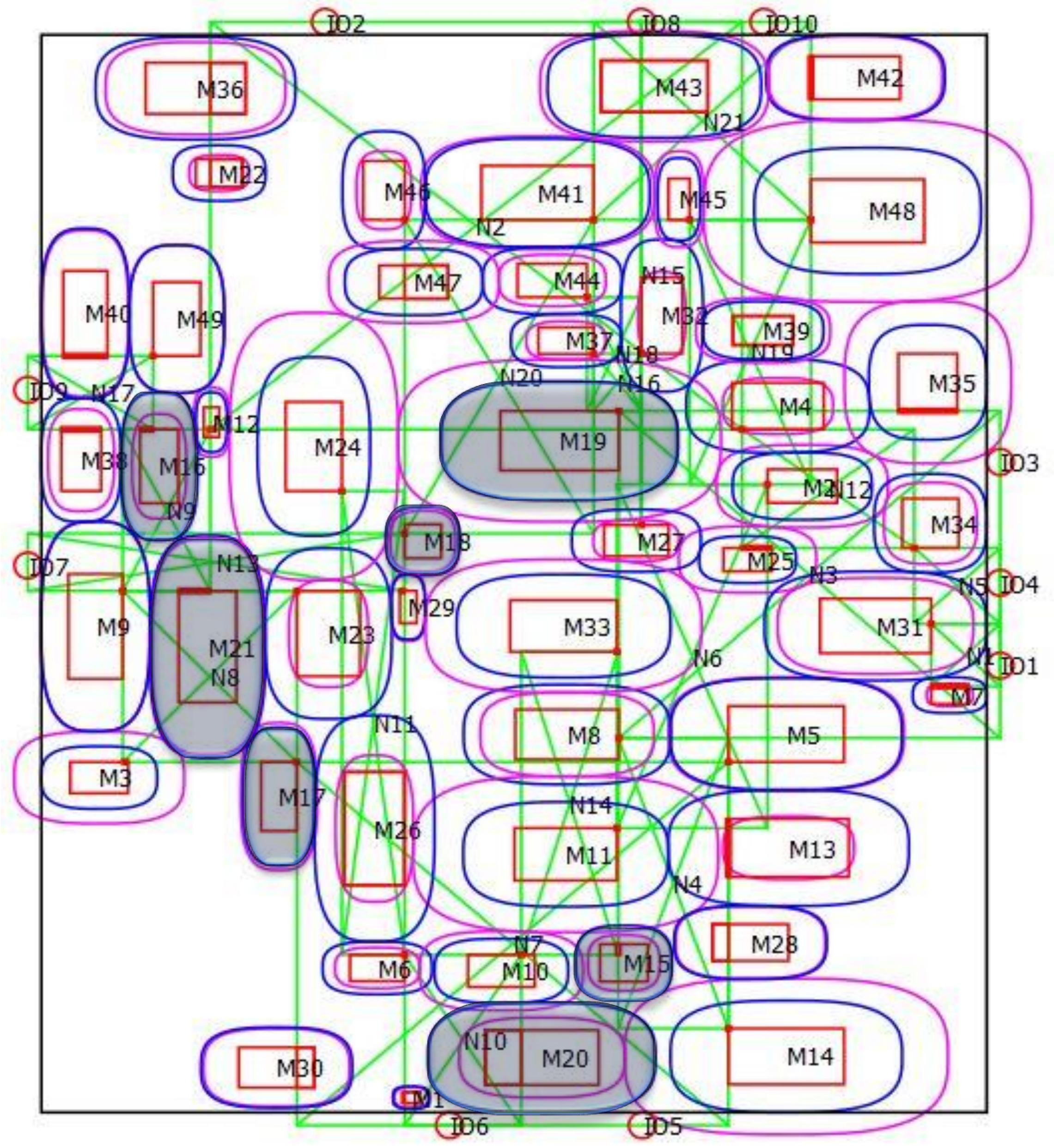

Figure 25: Dynamic-pin Multi-boundary Final-floorplan with relaxed boundaries.

4.2.2 Legalized Final-Floorplan

Again, as outlined in section 2.1.9, floorplanning with relaxed shapes will result in unresolved overlaps between the actual rectangular-shapes. To resolve these overlaps and legalize the floorplan we will, again, re-solve the final-floorplanning problem using macro-block nonoverlap constraints constructed from the macro-block rectangular boundaries and the I/O-block nonoverlap and abutment constraints constructed using the I/O-block square boundaries. Further, we initialize the optimization variables to the values produced by the final-floorplan with relaxed boundaries. Finally, we solve this floorplan legalization problem using IPOPT's warm-start capability in order to ensure that the resulting legalized floorplan does not stray too far from the floorplan produced using the relaxed boundaries.

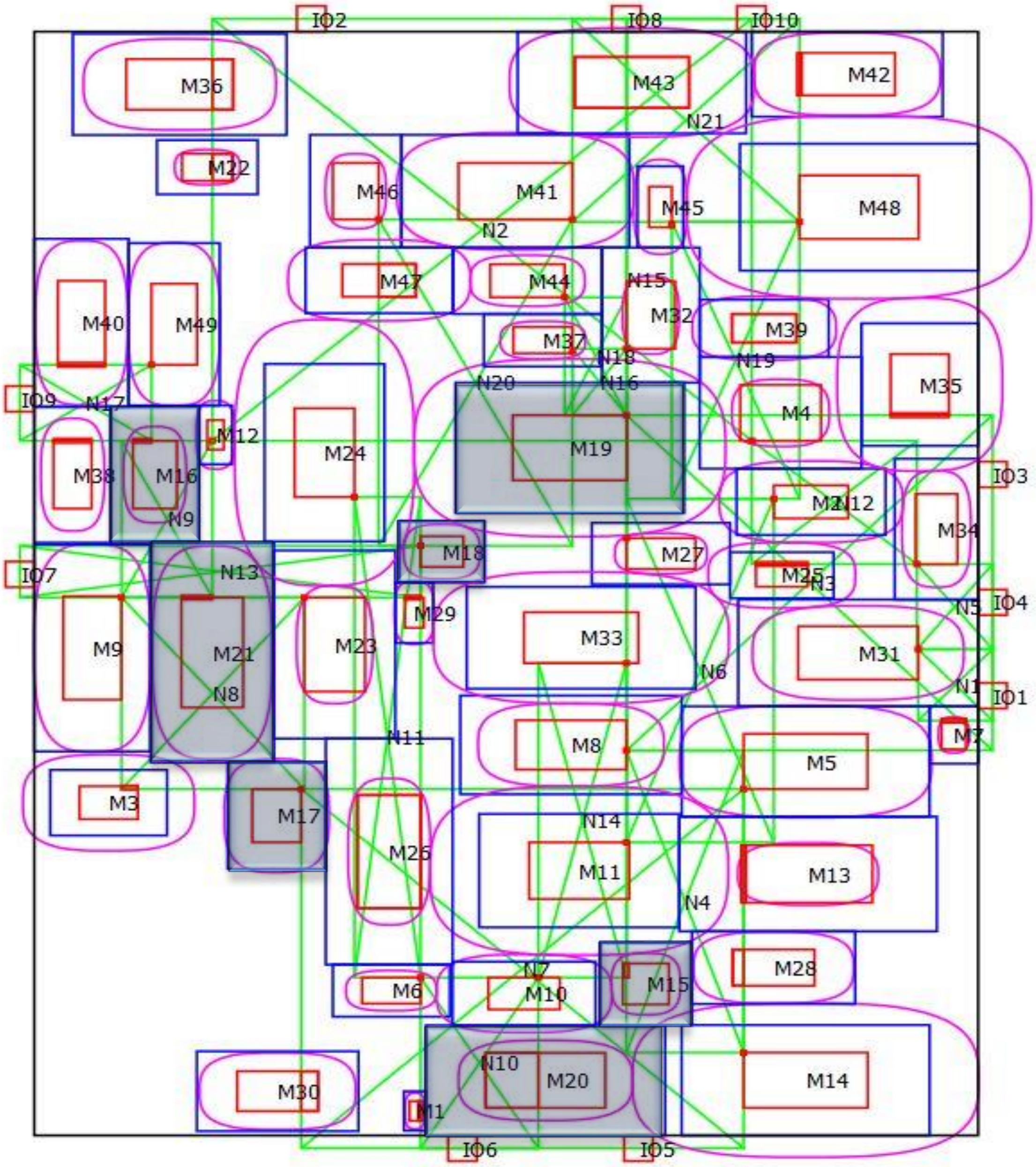

Figure 26: Dynamic-pin Multi-boundary Final-floorplan with rectangular physical-boundaries.

The resulting legalized final-floorplan is shown in *Figure 26* and has a total half-perimeter wire-length of 96.8 which is an 8.3% increase over legalized final-floorplan for the dynamic-pin single-boundary problem shown in *Figure 22*.

It should be noted that this floorplan in not very compact. However, this is not an entirely unexpected result as the main goal of this process was to minimize the total half-perimeter wire-length without any direct consideration for the size of the resulting floorplan. In fact, the only compaction being applied during this process is coming indirectly through the interaction of the web of pin containment constraints (constraint 4 and 5), the I/O-pin abutment constraints (constraint 7), and the physical-boundary containment constraints (constraint 6). While this process can produce relatively compact floorplans, as illustrated by the final floorplan seen in *Figure 22*, this is by no means guaranteed. Further, any modification to the final-floorplanning process which prioritizes floorplan size would virtually guarantee an increase in total half-perimeter wire-length as the price for the decrease in floorplan size, as has been repeatedly demonstrated during the generation of each of the compacted simplified-floorplans.

5 Conclusion

In this paper we have developed and implemented sequential nonlinear-programming methods for solving the thermal-aware soft-macro VLSI floorplanning with IO-block placement and a dynamic floorplan-boundary.

In particular, we proposed and developed a two-stage approach to this floorplanning problem. This consists of A simplified first-stage, which omits any consideration of the floorplan boundary and IO-block placement avoiding introducing adverse bias due to premature consideration of these elements, and a detailed second-stage, which uses information extracted from the first-stage to minimize or eliminate adverse bias when incorporating the floorplan boundary and IO-blocks back into the floorplanning process.

Additionally, we proposed and developed methods for providing and maintaining macro-block mobility throughout the optimization process allowing the floorplanner to consistently generate high quality floorplans. These methods include providing an initial layout to the optimizer which contains sufficient inter-block spacing to give the optimizer enough freedom to cluster macro-blocks belonging to the same net before the floorplan becomes too congested for blocks to easily move around each other. These methods also include approximating the macro-block physical-boundaries with curvilinear relaxations to allow macro-blocks to more easily slide around each other once the floorplan becomes congested and the macro-blocks come into contact with each other.

Further, we proposed and developed a novel method for making floorplanning thermally aware. Specifically, we modeled each macro-block’s thermal-signature as an additional macro-block boundary and then made the optimization process thermally aware by using those boundaries to formulate an additional set of nonoverlap constraints. This abstraction requires a minimal amount of effort to implement and execute while providing a very flexible approach for incorporating a substantial amount of thermal awareness in to the floorplanning process.

Finally, while this paper has developed and presented these methods in the context of soft-macro floorplanning with IO-block placement and a dynamic floorplan-boundary, these methods, and their underlying concepts, are based on intrinsic behaviors of the axis-aligned rectangle packing-problem and thus can, without expending significant effort, be adapted to form the foundation for the solution to wide range of VLSI floorplanning problems including those involving a mix of soft macro-blocks and both fixed and multiple-orientation hard macro-blocks.

References

[1] COIN-OR, "IPOPT," [Online]. Available: https://github.com/coin-or/Ipopt.

[2] Artelys, "Knitro," [Online]. Available: https://www.artelys.com/solvers/knitro/.

[3] nag, "Mathematical Optimization Software," [Online]. Available: https://www.nag.com/content/mathematical-optimization-software.

[4] PERFORCE, "IMSL Numerical Libraries," [Online]. Available: https://www.imsl.com/.

[5] A. Wächter and L. T. Biegler, "On the implementation of a primal-dual interior point filter line search alrogirthm for large-scale nonlinear programming," *Mathematical Programming,* vol. 106(1), pp. 25-57, 2006.

[6] Y. Nesterov, "A method of solving a convex programming problem with convergence rate O(1/k^2)," *Soviet Mathematics Doklady,* vol. 27, no. 2, pp. 372-376, 1983.

[7] J. Lu, P. Chen, C.-C. Chang, L. Sha, D. Huang, C.-C. Teng and C.-K. Cheng, "ePlace: Electrostatics based placement using Nesterov's method," in *ACM/EDAC/IEEE Design Automation Conference*, San Francisco, CA, 2014.

[8] C.-K. Cheng, A. B. Kahng, I. Kang and L. Wang, "RePlAce: Advancing Solution Quality and Routability Validation in Global Placement," *Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 38, no. 9, pp. 1717-1730, 2019.

[9] Y. Lin, Z. Jiang, J. Gu, W. Li, S. Dhar, R. Haoxing, B. Khailany and D. Z. Pan, "DREAMPlace: Deep Learning Toolkit-Enabled GPU Acceleration for Modern VLSI Placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems ,* vol. 40, no. 4, pp. 748-761, 2020.

[10] P. B. Morton, The Analytical Computational Geometry Library: A Computation Geometry Library for Scientists and Engineers, 2nd ed., Create Space, 2022.

[11] P. B. Morton, "Using an Analytical Computational-Geometry Libary to Model Nonoverlap and Bounday-Distance Constraints and their Application to Packing Poly-Bezier Shapes," *Optimization Online,* 29 December 2021.

[12] X. Li, K. Peng, F. Huang and W. Zhu, "PeF: Poisson's Equation Based Large-Scale Fixed-Outline Flooplanning," *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems,* vol. 42, no. 6, pp. 2002 - 2015, 2023.

[13] K. Peng and W. Zhu, "Pplace-MS: Methodologically Faster Poisson’s Equation-Based Mixed-Size Global Placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 43, no. 2, pp. 613-626, 2024.

[14] W. Zhu and H. Ai, "Poisson Energy Formulation for Floorplanning: Variational Analysis and Mathematical Foundations," *arXiv,* 9 October 2025.

[15] Y. Zhan, Y. Feng and S. S. Sapatnekar, "A fixed-die floorplanning algorithm using an analytical approach," in *Asia and South Pacific Conference on Design Automation*, Yokohama, Japan, 2006.

[16] W. C. Naylor, R. Donelly and L. sha, "Non-Linear Optimization System and Method for Wire Length," *US Patent 6301693,* 2001.

[17] MonarchIP, "ACGL," [Online]. Available: https://www.monarchip.com/.