Large-Scale Fixed-Outline Floorplanning Design Using Convex Optimization

Chaomin Luo, Student Member, IEEE, Miguel F. Anjos, Member, IEEE and Anthony Vannelli, Member, IEEE

Abstract—A two-stage optimization methodology is proposed to solve the fixed-outline floorplanning problem that is a global optimization problem for wirelength minimization. In the first stage, an attractor-repeller convex optimization model provides the relative positions of the modules on the floorplan. The second stage places and sizes the modules using second-order cone optimization. With the relative positions of modules obtained from the first stage, a Voronoi diagram is employed to obtain a planar graph and thus a relative position matrix to connect the two stages. An efficient approach to generate sparse relative position matrices and an interchange-free algorithm for local improvement of the floorplan are also presented in this paper. Overlap-free and deadspace-free floorplans are achieved in a fixed outline and floorplans with any specified percentage of whitespace can be produced. Experimental results on MCNC and GSRC benchmarks demonstrate that we obtain significant improvements on the best results in the literature for these benchmarks. Most importantly, our methodology provides greater improvement over other floorplanners as the number of modules increases.

Index Terms—Fixed-outline floorplanning, convex optimization, second-order cone programming, relative position matrix, wirelength minimization.

I. INTRODUCTION

Floorplanning is becoming increasingly important as a tool to design flows in hierarchical design [1]. Floorplanning dealing with fixed-die is fixed-outline floorplanning, while classical floorplanning handles variable-die. Fixed-outline floorplanning attempts to pack all the modules within a given fixed floorplan outline, and aims to simultaneously minimize wirelength and overlap, and possibly also timing [2]. It plays an important role in state-of-the-art hierarchical methods to multi-level fixed-die design of large scale ASICs and SoCs. However, addressing the requirements of fixed-outline floorplanning is significantly more difficult than for outline-free floorplanning [1], [2].

There have been many studies on floorplanning, most of which focus on area minimization for variable-die. Only recently did floorplanners begin tackling fixed-die floorplanning by minimizing the total wirelength [3]. Parquet [1] and Capo [4] are two typical floorplanners for dealing with fixed-die constraints when minimizing total wirelength. Adya and Markov [1] suggested a moving technique based on slack computation and simulated annealing (SA) to optimize the wirelength by using sequence-pair to represent the topology of a floorplan. The fixed outline is satisfied by using a better local search. Capo [4] developed a floorplacer that effectively combines min-cut placement with SA to form a fixed-outline floorplanner. At the stage of partitioning and placement, fixed-outline floorplanning is completed while taking routability into account.

Recently, several fixed-outline floorplanners have been developed based on SA. Chen et al. [5] developed a so-called IMF multilevel floorplanner using a two-phase technique. IMF consists of a top-down partitioning phase that partitions floorplan into subregions, and a bottom-up merging phase that merges subregions. The total wirelength is optimized via min-cut partitioning in the first phase and by SA-based fixed-die floorplanning implemented in each subregion in the second phase. The floorplanner IMF AFF incorporates the accelerative fixed-outline floorplanning (AFF) into IMF, and is 11 times faster than IMF but at the expense of an increase of 9% in the wirelength. Chen and Chang [6] proposed an adaptive fast-SA scheme based on B*-tree floorplan representation. The total wirelength can be effectively minimized by dynamically varying the weights of the objective function. Fixed-die floorplanning is achieved by taking outline constraints into account. Unlike SA-based fixed-outline floorplanners [1], [4], [5], [6], a fast geometric algorithm, called Traffic, has recently been proposed for wirelength minimization without SA [7]. Traffic also decomposes floorplanning into two steps due to its complexity. In the first step, the modules are grouped by local and global connectivity into several layers as rows by a partitioning algorithm. The modules in the same layer are moved to trapezoidal form by wirelength minimization in the second step. Squeezing the constructed rows transforms them into the required floorplans. The main advantage of this approach is the great improvement of runtime in comparison with an SA-based model.

Genetic algorithm (GA)-based fixed-outline floorplanners have been studied recently [8], [9]. Lin et al. [8] proposed a GA-based fixed-outline floorplanning to place all the modules in enlarged floorplan outlines. The loose outlines were enlarged to 115% of the total module area. Based on the
concept of the GA, Lin et al. [10] developed a genetic clustering algorithm for slicing floorplans considering fixed-outline and boundary constraints. Those modules that should have stronger connectivity with I/O pads are forced along the chip boundary to improve the total wirelength in their algorithm. Two evolutionary operations for searching the solution space to find a fixed-outline floorplan based on the GA also are described by Chen et al. [9].

Most recently, some other approaches were also proposed for fixed-outline floorplanning. For example, Liu et al. [11] achieved fixed-outline floorplanning by a local search method based on instance augmentation adopting a sequence-pair representation of the floorplan. In search of feasible and optimal solutions in the solution space, their algorithm explores both instances and sub-instances. However, it is difficult for their algorithm to attain zero-deadspace. Feng and Mehta [12] suggested a geometry-based moving approach for floorplanning design by minimizing the standard deviation of modules densities. Zhan et al. [13] proposed an analytical approach for fixed-outline floorplanning dealing with soft modules that minimizes total wirelength in two phases. The wirelength and area distribution density of modules are minimized in the first phase. The overlap area and wirelength are optimized in the second phase to achieve an overlap-free floorplan. Cong et al. [14] developed a floorplanner that uses recursive bipartitioning flow to minimize total wirelength and arranges soft and hard modules within a fixed-outline floorplan. Chen and Yoshimura [15] proposed a technique called Insertion After Remove (IAR) for solution perturbation for the SA and arrangement of modules in sequence pair. They suggested a new objective function with width, height and aspect ratio of the chip for fixed-outline floorplanning. This objective function improves the success rate and minimizes area and total wirelength simultaneously.

There have been few studies on fixed-outline floorplanning by convex optimization. Moh et al. [16] formulated the floorplanning problem as a geometric program and thus it is transformed into a convex optimization problem to exploit the advantage of convex optimization. Murata and Kuh [17] combined convex optimization with SA and a nonslicing floorplan representation including variable modules and preplaced modules to achieve zero-deadspace. However, their method to find a solution is very time-consuming.

In this paper, a two-stage convex-optimization-based method is proposed for fixed-outline floorplanning minimizing wirelength. An important property of convex optimization is that any local minimum is a global minimum for the problem. In the first stage, the relative positions of modules are obtained by minimizing an estimate of the total wirelength through an attractor-repeller (AR) convex optimization model. A relative position matrix (RPM) is then built from the solution to the first stage, and the modules are placed and sized in the second stage using a second convex model that minimizes rectilinear wirelength. This second stage is a convex optimization problem with linear and second-order cone (SOC) constraints. Overlap-free and deadspace-free floorplans are achieved in fixed-outline floorplanning. Our model also produces floorplans with any specified percentage of whitespace. To the best of our knowledge, this is the first time that a completely convex-optimization-based method is used for fixed-outline floorplanning. The proposed two-stage method is particularly suitable for fixed-outline floorplanning and can also be extended to classical floorplanning.

The rest of this paper is organized as follows. In Section II, the background and importance of zero-deadspace floorplanning is presented. Section III briefly describes the first stage model. Section IV discusses overlap and non-overlap constraints and explains a relative position matrix technique. Section V proposes a sparse relative position matrix using the Voronoi Diagram. Section VI presents the second stage convex optimization model. Afterward experimental results and comparisons with some other models are reported in Section VII. Finally, the important properties of the proposed methodology are summarized in Section VIII.

II. ZERO-DEADSPACE FIXED-OUTLINE FLOORPLANNING

The zero-deadspace (ZDS) floorplanning problem aims to pack a given set of soft modules inside a fixed-die floorplan without any deadspace and/or overlap among modules.

In modern VLSI design, deadspace-free and overlap-free fixed-outline floorplan is required because the size of fixed-die and package have usually been determined before the floorplanning design during the chip synthesis process. Therefore, it is necessary to pack a zero-deadspace and zero-overlap layout in the fixed die. Furthermore, top-level routing and pin assignment are iteratively performed in both global and detailed floorplanning with modern hierarchical design. Detailed floorplanning seeks ZDS and zero-overlap layout, in which there are no unused resources [2], [1]. Mosaic floorplan is a name used by Hong et al. [18] if the floorplanning achieves ZDS.

The methods for ZDS floorplanning include network-flow method ([19], [20]), partition-based method ([14]), and resistive network approach ([21]). Previous research work achieved ZDS on slicing floorplans ([22], [23], [14]) or on nonslicing floorplans ([21]). Some approaches allow general floorplans([19], [14]).

The first proposal to attain ZDS is due to Wimer et. al. [19] and Wang and Chen [21]. However, their algorithms did not consider the aspect ratios of soft modules in the floorplans. ZDS floorplans with area minimization were achieved by Wang and Chen [21] by transforming the nonslicing floorplanning problem into a resistive network problem. Wimer et. al. [19] described a network flow and planar graph approach that implements ZDS layout for area minimization and also the existence and uniqueness of a ZDS floorplan is proved.

Since that point, there was little research on ZDS until recent modern hierarchical design flows on ASIC and SoC become increasingly popular. ZDS is more regarded as a constraint instead of an objective in the formulation [1]. Kahng [2] questioned and challenged the supremacy of classical floorplanning and his work is the earliest suggestion that ZDS fixed-outline floorplanning is more consistent with the requirements of modern design. He developed a formulation
called Perfect Rectilinear Floorplanning Problem (PRFP) that produces provable ZDS perfectly packed rectilinear floorplans with the fixed-outline constraint. Actually, the ZDS floorplan is a compacted floorplan.

Some methods implement and achieve ZDS floorplans by experiments ([19], [20], [14]). Others proposed some theoretical analysis and potential results ([22], [23], [24], [18]). Peixoto et al. [22], Young and Wong [23] proposed upper bounds on the total area minimization by theoretical analysis. Mehta and Sherwani [24] considered a grid data representation of floorplanning and developed corresponding floorplanning algorithms by replacing module shape of rectilinear shapes such L-shapes by arbitrary rectilinear shapes. They stated that their algorithms are theoretically able to implement ZDS.

Two categories of methods are used to attain ZDS. Some methods directly achieve ZDS floorplanning, while others obtain ZDS floorplans by minimizing the whitespace inside floorplans. For example, the algorithm of Cong et al. [14] directly obtained a provable ZDS fixed-outline floorplanning solution by a recursive top-down area bi-partitioning algorithm. Conversely, whitespace is initially allowed in the floorplans in some algorithms. The iterative refinement and area redistribution approach is then used to achieve ZDS ([25]). The ZDS is potentially a result of their min-cut max-flow floorplanner. Feng et al. [20] introduced three sort of BFS-based (breadth first search) algorithms following the work of Kahng [2]. They used a min-cost max-flow network formulation that packs rectilinear-shaped modules in a ZDS floorplan. With advance of floorplanning design techniques during the past decade, theoretically, some floorplan representations can implement the compacted floorplan and thus a ZDS floorplan. For instance, CBL (Corner Block List) representation invented by Hong et al. [18] potentially achieves ZDS floorplans without empty rooms by assigning each room only one module in the slicing and nonslicing floorplans.

The ZDS floorplanning problem may be defined as follows: given \( n \) modules with areas \( a_i \) (\( i = 1, 2, ..., n \)) and the chip area \( A \) of the floorplan, the objective is to arrange all the modules inside the floorplan without any overlap, and minimize the area and/or the total wirelength such that \( \sum_{i=1}^{n} a_i = A = \bar{w}_F \times \bar{h}_F \), where \( \bar{w}_F \) and \( \bar{h}_F \) are the fixed width and height of the floorplan.

In our methodology, the sum of each area of modules is equal to the area \( A \) of the chip. We enforce this constraint into the second stage formulation by the height \( \bar{h}_F \) and width \( \bar{w}_F \) of the chip. We are capable of obtaining ZDS solution and also allowing any percentage of deadspace in the fixed-outline floorplan. One of main advantages of our methodology is that it is not restricted to slicing structures. The ZDS results for fixed-outline floorplanning are demonstrated using the MCNC and GSRC benchmarks.

### III. FIRST STAGE CONVEX OPTIMIZATION MODEL

In the first stage, we adopt the AR model which is based on a target distance concept [26]. Let each module \( i \) be represented by a circle of radius \( r_i \), where \( r_i \) is proportional to \( \sqrt{a_i} \), the square root of the area, \( a_i \), of module \( i \). The target distance for each pair of circles \( i, j \) is defined as \( t_{ij} := \sigma (r_i + r_j)^2 \), where \( \sigma > 0 \) is a parameter.

Furthermore, a generalized target distance is defined as \( T_{ij} := \frac{1}{c_{ij} + \varepsilon} \), where \( \varepsilon > 0 \) is a sufficiently small number, and \( c_{ij} \) is the connectivity between modules \( i \) and \( j \). The motivation for this definition can be found in [26]. To prevent circles from overlapping, the target distance is enforced via the objective function by introducing a penalty term which acts as a repeller: \( f\left(\frac{D_{ij}}{T_{ij}}\right) \), where \( f(z) = \frac{1}{z} - 1 \) for \( z > 0 \), and \( D_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2 \). The objective function is thus given by \( \min_{1 \leq i < j \leq n} c_{ij} D_{ij} + f\left(\frac{D_{ij}}{T_{ij}}\right) \). The interpretation is that the first term is an attractor that makes the two circles move closer together and pull them towards a layout where \( D_{ij} = 0 \), while the second term is a repeller that prevents the circles from overlapping. The parameter \( t_{ij} \) is a target value for the corresponding \( D_{ij} \). A piecewise function \( F_{ij} \), which is convex and continuously differentiable, is defined as

\[
F_{ij}(x_i, x_j, y_i, y_j) := \begin{cases} 
\frac{1}{2} D_{ij} + \frac{1}{2} t_{ij} - 1, & D_{ij} \geq T_{ij} \\
\frac{1}{2} t_{ij} t_{ij} - 1, & 0 \leq D_{ij} < T_{ij}
\end{cases}
\]

where \( D_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2 \). Let \( (x_i, y_i) \) and \( (x_j, y_j) \) denote the coordinates of the centers of modules \( i \) and \( j \).

The AR model in the first stage is:

\[
\min_{(x_i, y_i), \bar{w}_F, \bar{h}_F} \sum_{1 \leq i < j \leq n} F_{ij}(x_i, x_j, y_i, y_j) - K \ln \left(\frac{D_{ij}}{T_{ij}}\right)
\]

s.t.

\[
\begin{align*}
x_i + r_i &\leq \frac{1}{2} \bar{w}_F, \quad \forall i \\
r_i - x_i &\leq \frac{1}{2} \bar{w}_F, \quad \forall i \\
y_i + r_i &\leq \frac{1}{2} \bar{h}_F, \quad \forall i \\
r_i - y_i &\leq \frac{1}{2} \bar{h}_F, \quad \forall i \\
\bar{w}_F^{low} &\leq \bar{w}_F \leq \bar{w}_F^{up}, \\
\bar{h}_F^{low} &\leq \bar{h}_F \leq \bar{h}_F^{up}
\end{align*}
\]

where \( \bar{h}_F, \bar{w}_F \) are the height and width of the floorplan; \( \bar{h}_F^{low}, \bar{h}_F^{up}, \bar{w}_F^{low}, \) and \( \bar{w}_F^{up} \) are the lower and upper bounds of the height and width of the floorplan, respectively; and \( K \) is a parameter selected empirically.

Without the term \(-K \ln \left(\frac{D_{ij}}{T_{ij}}\right)\) in formulation (1), it was proven that this problem is convex [27]. By solving formulation (1), the solution of the first stage provides relative positions within the floorplan for all the modules.

### IV. NON-OVERLAP CONSTRAINTS AND RELATIVE POSITION MATRIX

If \( (x_i, y_i), (x_j, y_j), w_i, w_j, h_i \) and \( h_j \) are the coordinates, widths and heights of two rectangular modules, the non-overlap constraints for each pair of modules can be expressed as

\[
\begin{align*}
\frac{1}{2}(w_i + w_j) &\leq |x_i - x_j| & \text{if } |y_i - y_j| \leq \frac{1}{2}(h_i + h_j), \\
\frac{1}{2}(h_i + h_j) &\leq |y_i - y_j| & \text{if } |x_i - x_j| \leq \frac{1}{2}(w_i + w_j).
\end{align*}
\]
These constraints can be expressed as separation in the $x$-direction:

$$0 \geq \frac{1}{2}(w_i + w_j) - |x_i - x_j|, \tag{3}$$

or separation in the $y$-direction:

$$0 \geq \frac{1}{2}(h_i + h_j) - |y_i - y_j|. \tag{4}$$

It is noted that the constraints (3-4) are disjunctive, non-linear, and non-convex. Once relative positions for the modules have been obtained from the first stage, we may remove the absolute values from inequalities (3) and (4), thus linear constraints are obtained.

The relative positions are encoded in a RPM: if module $i$ is horizontally separated from module $j$, “1” is used to represent this case as an entry in RPM, meaning there is only one constraint separating in the $x$-direction as (3).

We use “11” to express that module $i$ is on the left of module $j$ (Fig. 1A). The following inequality is satisfied:

$$0 \geq \frac{1}{2}(w_i + w_j) - (x_i - x_j). \tag{5}$$

“12” is used to denote that module $i$ is on the right of module $j$ (Fig. 1B). The following inequality is satisfied:

$$0 \geq \frac{1}{2}(w_i + w_j) - (x_i - x_j). \tag{6}$$

If module $i$ is vertically separated from module $j$, “2” is used to represent this case, in which there is only one constraint separating in the $y$-direction as (4). We use “21” to express that module $i$ is above module $j$ (Fig. 1C). The following inequality is satisfied:

$$0 \geq \frac{1}{2}(h_i + h_j) - (y_i - y_j). \tag{7}$$

“22” is used to express that module $i$ is below module $j$ (Fig. 1D). The following inequality is satisfied:

$$0 \geq \frac{1}{2}(h_i + h_j) - (y_j - y_i). \tag{8}$$

If we have the relative positions of two modules separated diagonally (Fig. 2), a rule is defined to determine how the two modules are separated. If $\Delta y \geq \Delta x$, then these two modules should be separated in the $y$-direction axis. If $\Delta x > \Delta y$, then these two modules should be separated in the $x$-direction.

In summary, in the RPM, the following entries represent the relative position relations of two modules:

- “11” with index $(i, j)$: module $i$ is horizontally separated from module $j$, and module $i$ is on the left of module $j$;
- “12” with index $(i, j)$: module $i$ is horizontally separated from module $j$, and module $i$ is on the right of module $j$;
- “21” with index $(i, j)$: module $i$ is vertically separated from module $j$, and module $i$ is above module $j$;
- “22” with index $(i, j)$: module $i$ is vertically separated from module $j$, and module $i$ is below module $j$.

V. SPARSE RELATIVE POSITION MATRIX AND VORONOI DIAGRAM

We now demonstrate the contribution of the RPM and of a sparse version of it. The motivation for using a sparse relative position matrix (SRPM) instead of the full RPM is to have fewer constraints for the second stage model, and thus faster computation of the solution. The one-dimensional six-module case given in Fig. 4 is used to show how the RPM and SRPM are built. With the notation above, the RPM for Fig. 4 is shown in (9).

$$RPM = \begin{pmatrix}
0 & 11 & 11 & 12 & 11 & 11 \\
0 & 12 & 12 & 12 & 12 & 12 \\
0 & 12 & 12 & 11 & 0 & 11 \\
0 & 11 & 11 & 0 & 11 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix} \tag{9}$$

The distances between modules are obtained from the relative position of modules and included in a distance matrix (DM), which indicates the intervals among modules. The DM for the one-dimension six-module case in Fig. 4 is constructed based on the distance between modules shown in (10). A parameter $\delta$ is defined to discriminate among the distances between modules. Those modules with distance at least $\delta$ from any given module are not significant for the relative position of the module. If module 1, for instance, is the current module in Fig. 4, and $\delta = 2$, its neighboring modules within a distance of $\delta$ are modules 3, 4, and 5. Modules 2 and 6 are located farther than $\delta$ from module 1 thus the relative positions between module 1 and module 2, as well as between module

![Fig. 1. Relative positions of two modules separated horizontally and vertically.](image)

![Fig. 2. Relative position of two modules separated diagonally.](image)
For instance, DM(2,4) = 5 denotes that the interval between module 2 and module 4 is 5. It is only necessary to record the relative position if two modules have distance less than 3. Therefore, the element in the SRPM corresponding to (2,4) is recorded to be zero, i.e., SRPM(2,4):=0.

By replacing those entries by zeros, a sparse DM (SDM) is formed as (11). The RPM shown in (9) becomes the SRPM in (12) by substituting less significant entries of RPM by zeros. For instance, DM(2,4) = 5 denotes that the interval between module 2 and module 4 is 5. It is only necessary to record the relative position if two modules have distance less than 3. Therefore, the element in the SRPM corresponding to (2,4) is recorded to be zero, i.e., SRPM(2,4):=0.

An efficient approach of building a sparse RPM for the two-dimensional case is illustrated using the circuit ami33. Based on the solution from the first stage, a VD is illustrated in Fig. 5A. In the two-dimensional case, the distance δ=1 represents that the central module has relationship with its one-layer neighboring modules. Similarly, distance δ=2 indicates that the central module is closely related to its two-layer neighboring modules in Fig. 5A. Those modules beyond distance δ from any given central module are not significant in terms of relative position between modules. In Fig. 5, the central module 14 is closely related to its first-layer adjacent modules. The rest of modules which are farther than the first-layer of the module 14 is not crucial for the relative position.

Every module is shown to be adjacent to other modules. Module 10, for instance, is adjacent to modules 9, 11, 12, 13, and 19 in the VD. Module 14 is adjacent to underlined modules 1, 6, 15, 16, 19, and 21 (Fig. 5A). Module 24 is adjacent to modules 1, 15, 22, 23, 25, and 27. After the layout problem is solved (Fig. 5B), module 10 is adjacent to module 9, 12, 13, 19 and 21 in the VD. Module 14 is adjacent to underlined modules 1, 6, 15, 19, and 21. Module 24 is adjacent to modules 1, 15, 22, 23, and 24.

The neighborhood matrix (NM) for ami33 contains elements with 1’s, for instance NM(1,4)=1, NM(6,14)=1, NM(14,15)=1, NM(14,16)=1, NM(14,19)=1, NM(14,21)=1, and the rest of elements of NM is zero. As NM and SRPM are matrices with same size. Then every element with 1 in NM is mapped into the SRPM where the elements are all zeros. The rest of elements of the SRPM is number such as 11, 12, 21, and 22 that denote different relative positions between two modules depicted in Section IV. There-
fore, accordingly, SRPM(1,14), SRPM(6,14), SRPM(14,15),
SRPM(14,16), SRPM(14,19), and SRPM(14,21) are all zero.

VI. SECOND STAGE CONVEX OPTIMIZATION MODEL

So far, the relative positions of modules has been obtained
from the first stage. In the second stage, we use a Second
Order Cone Programming (SOCP) formulation to minimize
wirelength and obtain deadspace-free and overlap-free floor-
plans.

SOCOP is a special class of convex programming problems,
in which a linear objective function is minimized subject to
SOC constraints. As a special case of convex optimization,
SOCOPs have attracted much attention [33]. SOCOP problems
can be efficiently solved by specialized interior-point methods.
Additionally, the disjunctive constraints in (3) and (4) are
replaced by linear constraints using the SRPM.

Also known as quadratic, ice-cream or Lorentz cone, a
standard SOC of dimension \(n\) is expressed as [33]

\[
\mathcal{E}_n = \left\{ \begin{bmatrix} u \\ t \end{bmatrix} \mid u \in \mathbb{R}^{n-1}, t \in \mathbb{R}, \|u\| \leq t \right\}, \tag{14}
\]

where \(u \in \mathbb{R}^{n-1}, t \in \mathbb{R}\), and the SOC defines a convex set.
For \(n=3\), the SOC is illustrated in Fig. 6.

A SOC constraint of dimension \(n\) is defined as

\[
\|Ax + b\| \leq c^T x + d \iff \begin{bmatrix} A \\ c^T \end{bmatrix} x + \begin{bmatrix} b \\ d \end{bmatrix} \in \mathcal{E}_n, \tag{15}
\]

where \(x \in \mathbb{R}^k\), parameter \(A \in \mathbb{R}^{(n-1) \times k}\), \(b \in \mathbb{R}^{n-1}\), \(c \in \mathbb{R}^k\),
\(d \in \mathbb{R}\). Therefore, an SOCOP is a convex optimization problem
of the form:

\[
\begin{align*}
\min & \quad g^T x \\
\text{s.t. } & \quad \|A_i x + b_i\| \leq c_i^T x + d_i, \; \forall \; i = 1, ..., L
\end{align*} \tag{16}
\]
where, the optimization variable is \( x \in \mathbb{R}^n \), \( g \in \mathbb{R}^n \) is the objective function, \( A_i \in \mathbb{R}^{(n_i-1) \times n}, b_i \in \mathbb{R}^{n_i-1}, c_i \in \mathbb{R}^n, d_i \in \mathbb{R} \). This SOCP problem may be converted into standard form by introducing variables \( u^i \in \mathbb{R}^{n_i-1} \) and \( t^i \in \mathbb{R} \) as follows:

\[
\begin{align*}
\min & \quad g^T x \\
\text{s.t.} & \quad \|u^i\| \leq t^i, \quad \forall \ i = 1, \ldots, L \\
& \quad u^i = A_i x + b_i, \quad \forall \ i = 1, \ldots, L \\
& \quad t^i = c_i^T x + d_i, \quad \forall \ i = 1, \ldots, L,
\end{align*}
\]

where \( \|u\| \leq t \) is the standard SOC constraint. In the following sections, we attempt to formulate area and aspect ratio constraints as SOC constraints.

A. Area Constraints

The area constraint, \( w_i h_i = a_i \), for each soft module can be relaxed as \( w_i h_i \geq a_i \). For \( a_i > 0 \), this area constraint can be formulated as an SOC constraint:

\[
\begin{align*}
& \quad w_i h_i \geq a_i \\
\iff & \quad h_i^2 + 2w_i h_i + w_i^2 \geq (h_i - w_i)^2 + 4a_i \\
\iff & \quad h_i + w_i \geq \sqrt{(h_i - w_i)^2 + (2a_i)^2} \\
\iff & \quad h_i + w_i \geq \|h_i - w_i\|_2, \quad \forall \ i.
\end{align*}
\]

B. Aspect Ratio Constraints

The aspect ratio \( \beta_i \) for module \( i \) is defined as \( \beta_i = \max\{h_i/w_i\} / \min\{h_i/w_i\} \). To avoid excessively narrow modules in either direction in the floorplan, we assume that the aspect ratio of module \( i \) must be bounded above by a given value \( \beta_i^* > 0 \).

If given \( w_i^{low} = h_i^{low} = \sqrt{a_i/\beta_i^*} \), where \( a_i = w_i h_i \), then \( w_i \geq w_i^{low} > 0, w_i^2 \geq a_i/\beta_i^*, \beta_i^* w_i^2 \geq a_i, \beta_i^* \geq h_i/w_i \). Similarly, as \( h_i \geq h_i^{low} > 0 \), therefore, \( \beta_i^* \geq h_i/w_i \). With inequality \( \beta_i^* \geq h_i/w_i \) and \( w_i h_i = a_i \), we obtain \( \beta_i^* \geq h_i^2/a_i \). Further, we obtain \( a_i \beta_i^* \geq h_i^2 \). Combining inequality \( \beta_i^* \geq h_i^2/a_i \) yields \( \beta_i^* \geq w_i^2/a_i \). Therefore, \( a_i \beta_i^* \geq w_i^2 \).

The aspect ratio constraint for the height of every module denoted by inequality \( a_i \beta_i^* \geq h_i^2 \) can be formulated by an SOC cone constraint as follows:

\[
\begin{align*}
& \quad a_i \beta_i^* \geq h_i^2 \\
\iff & \quad a_i^2 + 2a_i \beta_i^* + \beta_i^* \geq a_i^2 - 2a_i \beta_i^* + \beta_i^* \geq 4a_i \\
\iff & \quad (a_i + \beta_i^*)^2 \geq (a_i - \beta_i^*)^2 + (2h_i)^2 \geq 0 \\
\iff & \quad a_i + \beta_i^* \geq \sqrt{(a_i - \beta_i^*)^2 + (2h_i)^2} \\
\iff & \quad a_i + \beta_i^* \geq \sqrt{2h_i} \|a_i - \beta_i^*\|_2, \quad \forall \ i.
\end{align*}
\]
can be similarly formulated by an SOC constraint as follows:
\[ a_i \beta_i^* \geq w_i^2 \]
\[ \iff a_i^2 + 2a_i \beta_i^* + \beta_i^* \geq a_i^2 - 2a_i \beta_i^* + \beta_i^* + 4w_i^2 \]
\[ \iff (a_i + \beta_i^*)^2 \geq (a_i - \beta_i^*)^2 + (2w_i)^2 \geq 0 \]
\[ \iff a_i + \beta_i^* \geq \sqrt{(a_i - \beta_i^*)^2 + (2w_i)^2} \]
\[ \iff a_i + \beta_i^* \geq \left\| \frac{a_i - \beta_i^*}{2w_i} \right\|, \forall i. \] (20)

By incorporating the SOC constraints above for the area and aspect ratio of the modules, the complete problem of minimizing the total wirelength for fixed-outline floorplanning is formulated as:
\[
\min_{(x_i, y_i, w_i, h_i)} \sum_{1 \leq i < j \leq n} c_{ij}(u_{ij} + v_{ij})
\]
\[
\text{s.t.} \quad u_{ij} \geq x_i - x_j, \\
\quad u_{ij} \geq x_j - x_i, \\
\quad v_{ij} \geq y_i - y_j, \\
\quad v_{ij} \geq y_j - y_i, \\
\quad x_i + \frac{1}{2} w_i \leq \frac{1}{2} \bar{w}_F \quad \forall i, \\
\quad y_i + \frac{1}{2} h_i \leq \frac{1}{2} \bar{h}_F \quad \forall i, \\
\quad \frac{1}{2} w_i - x_i \leq \frac{1}{2} \bar{w}_F \quad \forall i, \\
\quad \frac{1}{2} h_i - y_i \leq \frac{1}{2} \bar{h}_F \quad \forall i, \\
\quad w_{i}^{\text{low}} \leq w_i \leq w_{i}^{\text{up}} \quad \forall i, \\
\quad h_{i}^{\text{low}} \leq h_i \leq h_{i}^{\text{up}} \quad \forall i, \\
\quad \frac{\left\| \frac{h_i - w_i}{2 \sqrt{a_i}} \right\|}{2} \leq h_i + w_i \quad \forall i, \\
\quad \frac{\left\| \frac{a_i - \beta_i^*}{2h_i} \right\|}{2} \leq a_i + \beta_i^* \quad \forall i, \\
\quad \frac{\left\| \frac{a_i - \beta_i^*}{2w_i} \right\|}{2} \leq a_i + \beta_i^* \quad \forall i,
\]

where \(1 \leq i < j \leq n\), and \(\bar{w}_F \) and \(\bar{h}_F \) are the fixed width and height of the floorplan, and \((u_{ij} + v_{ij})\) is the rectilinear distance \(|x_i - x_j| + |y_i - y_j|\) between modules \(i\) and \(j\).

Now we obtain a convex SOC model that can be efficiently solved using the commercially available conic software package MOSEK [34]. Note that the non-overlap constraints are not shown in (21). They are included in this formulation in the form of linear constraints derived from the SRPM.

\section*{C. Algorithm for Interchange-free Local Improvement}

Ordinarily, when we solve formulation (21), we can achieve a deadspace-free and overlap-free floorplan subject to the aspect ratios of modules. It is possible that (21) be infeasible, however. If this happens, we solve (21) without any aspect ratio constraints. After this, we find the modules with aspect ratios violating the required upper bound, and locally relax the non-overlap constraints for those modules (call them central modules). More precisely, we select two-layer modules surrounding the central module and relax the relationships of relative positions between the central modules and surrounding modules by putting zero to the entries in RPM. We solve the SOCP problem iteratively until the upper bound of aspect ratios satisfies the required value. Several definitions are depicted as follows and the algorithm to implement this procedure is stated as Algorithm 1.

**Definition 1: (Central Modules \(M_i\))** Central Modules \(M_i(i = 1, 2, \ldots, p)\) are those modules whose aspect ratios exceed the required upper bound.

**Definition 2: (First Layer Modules \(M_{ij}\))** First Layer Modules \(M_{ij}(i = 1, 2, \ldots, p; j = 1, 2, \ldots, q)\) are those modules immediately adjacent to the Central Modules \(M_i\).

**Definition 3: (Second Layer Modules \(M_{ijk}\))** Second Layer Modules \(M_{ijk}(i = 1, 2, \ldots, p; j = 1, 2, \ldots, q; k = 1, 2, \ldots, r)\) are all the modules which are immediately adjacent to the first layer modules \(M_{ij}\).

**Definition 4: (Relaxed RPM)** The relaxed RPM is the RPM whose entries with the row and column indices in the RPM denoting the Modules \(M_i, M_{ij}\) and \(M_{ijk}\), are set to be zeros. In other words, it relaxes the relationships of the relative positions between \(M_i\) and \(M_{ij}/M_{ijk}\).

As an example we solved (21) for ami49 and the layout obtained is shown in Fig. 7, namely, first round layout. In this layout, all aspect ratio values satisfy the upper bound constraint, i.e., they are less than \(\beta_i=10\), except that the aspect ratio of Module 21 is 19.0408.

According to the description of Algorithm 1 and Definitions 3 and 4, the Central Module whose aspect ratio dissatisfy the upper bound constraint is \(M_{21}\). The first layer modules adjacent to the central module \(M_{21}\) are \(M_{1}, M_{13}, M_{22}, M_{24}, M_{39},\) and \(M_{45}\). They are represented as \(M_{21,1}, M_{21,13,21,22,21,24,21,39,}\) and \(M_{21,45}\), respectively.

[Fig. 7. First round layout for ami49 circuit.]

The successful layout is found by re-solving the SOCP problem as shown in Fig. 9.

**VII. EXPERIMENTAL RESULTS**

We use the clique model for transforming hypergraphs to two-pins nets and the half-perimeter wirelength (HPWL) to measure the quality of all our floorplans. For a set of modules with total area \(A\) and maximum whitespace fraction \(\gamma\), as well
Algorithm 1: Algorithm for Interchange-free Local Improvement

**Input:** SRPM

**Output:** Aspect ratios, module dimensions

1. Solve SOCP model without aspect ratio constraints;
2. If all the aspect ratios are satisfactory, goto Step 9; otherwise, goto Step 3;
3. Select all the Central Module $M_i$;
4. Compute and select all the First Layer Modules $M_{ij}$;
5. Compute and select all the Second Layer Modules $M_{ijk}$;
6. Set up relaxed SRPM;
7. With the relaxed SRPM, solve the SOCP model without aspect ratio constraints to obtain a layout with overlaps; based on this relative position to update RPM;
8. Re-solve SOCP model with aspect ratio constraints;
9. End.

![Fig. 8. Second round layout for ami49, where the relative positions of the first and second layer modules are locally relaxed.](image)

![Fig. 9. Final layout by the third iteration for ami49](image)

as given aspect ratio $\alpha$ for a fixed outline, the height $H_F$ and width $W_F$ of the chip can be given as follows [1]:

$$H_F = \sqrt{(1 + \gamma)A} \alpha \quad W_F = \sqrt{(1 + \gamma)A/\alpha}. \quad (22)$$

The proposed methodology is applied to the standard MCNC and GSRC benchmarks to demonstrate its effectiveness and flexibility. In this section, our methods are compared with several state-of-the-art academic floorplanners: Murata and Kuh (MK) [17], Adya and Markov (AM) [1], Parquet [1], [35], Capo [4], IMF and IMFAFF [5], and a commercial tool, TimberWolf 1.3.3 [36], [37]. All reported wire lengths are measured using the HPWL, which was also used by [17], [1] and [37].

All the modules are chosen to be soft modules with fixed areas and variable dimensions, and (as an approximation) all the pins are assumed to be at the center of the modules. The multi-pin nets are transformed into cliques using the clique model.

The first stage was solved using the optimization package MINOS [38] via the modeling language AMPL [39] accessed on the NEOS server [40]. The second stage was solved by MOSEK [34] on a Linux SUN Fire-V890 server with 16 1200-MHz processors and 32 GB RAM. We report only CPU time for the second stage. Solving the first stage takes significantly less time than the second stage. Therefore, the CPU time for the first stage is negligible. The final total wirelength was computed by HPWL to compare the quality of the floorplans obtained. For these instances of fixed-outline floorplanning, we respect the dimensions of the floorplans provided in the MCNC and GSRC benchmarks, so that the fixed layout area is a sum of the area of all of the modules included in a circuit if there are no whitespace. We chose $\alpha = 1$ in (22).

### A. Detailed Experimental Results for the MCNC and GSRC Benchmarks

We solved the first stage formulation (1) and second stage formulation (21) bridged by VD. Each circuit was run once, and we report the obtained floorplan with CPU time and HPWL wirelength in Table I for MCNC benchmark and Table II.

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Total Area</th>
<th>Our Methodology</th>
<th>Runtime</th>
<th>HPWL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(mm$^2$)</td>
<td>(mm$^2$)</td>
<td>(s)</td>
<td>(mm)</td>
</tr>
<tr>
<td>apte</td>
<td>46.56</td>
<td>46.56</td>
<td>1.03</td>
<td>39.7</td>
</tr>
<tr>
<td>xerox</td>
<td>19.35</td>
<td>19.35</td>
<td>1.12</td>
<td>41.1</td>
</tr>
<tr>
<td>hp</td>
<td>8.30</td>
<td>8.30</td>
<td>1.29</td>
<td>14.2</td>
</tr>
<tr>
<td>ami33</td>
<td>1.16</td>
<td>1.16</td>
<td>6.06</td>
<td>50.08</td>
</tr>
<tr>
<td>ami49</td>
<td>35.4</td>
<td>35.4</td>
<td>9.96</td>
<td>69.9</td>
</tr>
</tbody>
</table>

### B. Comparison with the Model of MK

First, we compare our approach with the model of Murata and Kuh (MK) [17]. Recall that Murata and Kuh employed a convex optimization algorithm to optimize the aspect ratios of the soft modules, and iteratively carried out this procedure by a simulated annealing algorithm to improve the sequence-pair for floorplanning.

Note that while in Murata and Kuh’s experiment, the chip aspect ratio for every benchmark was modified to 1, thus requiring the chip to be square in shape, the dimensions of
the floorplans in our experiments comply with the original ones provided in the MCNC benchmark. Like us, Murata and Kuh also assumed that every module is soft with aspect ratio being constrained to lie in the same range of 0.1 to 10.

Table III lists the areas, computation times, and total wire lengths reported by Murata and Kuh. The comparisons with regard to wirelength and runtime are reported in Table IV. The results show that our method requires much less CPU time than Murata and Kuh’s method. Furthermore, the HPWL is competitive and in fact we obtain an improvement in wirelength over MK for the two largest benchmark problem, ami33 and ami49. On average, our method is orders of magnitude faster, while wirelength is only 4.24% worse than the simulated annealing based results reported by Murata and Kuh.

### C. Comparison with the Model of AM

We also compare our results with the results reported by Adya and Markov (AM) [1]. Recall that Adya and Markov used sequence-pair to represent the topology of a floorplan, together with a moving technique based on slack computation and simulated annealing.

### TABLE II

<table>
<thead>
<tr>
<th>GSRC Experimental results with our model</th>
<th>Our Methodology</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total area ($mm^2$)</td>
<td>(mm)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>circuit</td>
<td>Our area ($mm^2$)</td>
<td>Runtime (s)</td>
<td>HPWL (mm)</td>
<td></td>
</tr>
<tr>
<td>n10a</td>
<td>22.17</td>
<td>22.17</td>
<td>1.21</td>
<td>37090</td>
</tr>
<tr>
<td>n30a</td>
<td>20.86</td>
<td>20.86</td>
<td>5.96</td>
<td>117236</td>
</tr>
<tr>
<td>n50a</td>
<td>19.86</td>
<td>19.86</td>
<td>10.08</td>
<td>123860</td>
</tr>
<tr>
<td>n100a</td>
<td>17.95</td>
<td>17.95</td>
<td>55.89</td>
<td>197490</td>
</tr>
<tr>
<td>n200a</td>
<td>17.57</td>
<td>17.57</td>
<td>1026.43</td>
<td>356520</td>
</tr>
<tr>
<td>n300a</td>
<td>27.32</td>
<td>27.32</td>
<td>1654.70</td>
<td>477800</td>
</tr>
</tbody>
</table>

### TABLE III

<table>
<thead>
<tr>
<th>RESULTS REPORTED BY MK</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MCNC circuit</td>
<td>Total area ($mm^2$)</td>
<td>Area ($mm^2$)</td>
<td>Runtime (s)</td>
<td>HPWL (mm)</td>
</tr>
<tr>
<td>n10a</td>
<td>46.56</td>
<td>46.55</td>
<td>789</td>
<td>344.36</td>
</tr>
<tr>
<td>n30a</td>
<td>19.35</td>
<td>19.50</td>
<td>1198</td>
<td>401.25</td>
</tr>
<tr>
<td>n50a</td>
<td>8.30</td>
<td>8.83</td>
<td>1346</td>
<td>118.82</td>
</tr>
<tr>
<td>n100a</td>
<td>1.16</td>
<td>1.16</td>
<td>75684</td>
<td>53.39</td>
</tr>
<tr>
<td>n200a</td>
<td>35.4</td>
<td>35.58</td>
<td>612103</td>
<td>775.10</td>
</tr>
</tbody>
</table>

### TABLE IV

<table>
<thead>
<tr>
<th>IMPROVEMENTS IN RUNTIME AND WIRE LENGTH COMPARED WITH MK</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MCNC circuit</td>
<td>Our Methodology vs MK [17]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Speed-up (avg)</td>
<td>WL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>apte</td>
<td>766</td>
<td>-15.29%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>xerox</td>
<td>1070</td>
<td>-2.43%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hp</td>
<td>1043</td>
<td>-19.51%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ami33</td>
<td>12489</td>
<td>+6.20%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ami49</td>
<td>61456</td>
<td>+9.82%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>15365</td>
<td>-4.24%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### TABLE V

<table>
<thead>
<tr>
<th>RESULTS REPORTED BY AM</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MCNC circuit</td>
<td>Total area ($mm^2$)</td>
<td>Area (mm$^2$)</td>
<td>Runtime avg (s)</td>
<td>WL min/avg (mm)</td>
</tr>
<tr>
<td>apte</td>
<td>46.36</td>
<td>46.97/48.95</td>
<td>15.4</td>
<td>464/560</td>
</tr>
<tr>
<td>xerox</td>
<td>19.35</td>
<td>19.51/20.62</td>
<td>20.1</td>
<td>373/468</td>
</tr>
<tr>
<td>hp</td>
<td>8.30</td>
<td>8.96/9.72</td>
<td>15.3</td>
<td>177/214</td>
</tr>
<tr>
<td>ami33</td>
<td>1.16</td>
<td>1.18/1.24</td>
<td>31.0</td>
<td>62.57/75.4</td>
</tr>
<tr>
<td>ami49</td>
<td>35.4</td>
<td>36.07/37.8</td>
<td>31.9</td>
<td>673/812</td>
</tr>
</tbody>
</table>

Table V summarizes the areas, computation times, total wire lengths reported by Adya and Markov. The comparisons with regard to wire length and runtime are reported in Table VI. The results show that our method requires less CPU time compared to Adya and Markov’s method. Furthermore, our average HPWL is consistently better, and the best floorplan improves on their best floorplan for several benchmarks. On average, our HPWL wirelength is 24.49% better than theirs with 10.62 times less CPU time than their model.

### D. Comparison with the TimberWolf

Then, we compare our approach with a commercial tool, TimberWolf1.3.3, which is based on simulated annealing, to further validate our model and results. The placement results for benchmark MCNC of TimberWolf1.3.3 ([36]) was reported in [37]. Cong et al [37] generated TimberWolf placement results by ignoring the routing space, which is greatly suitable for our solution to compare with TimberWolf since our floorplanner does not currently take routing space requirements into account, neither.

Table VII lists the areas, computation times, and total wire lengths by running a commercial tool TimberWolf1.3.3. The comparisons with regard to wire length and runtime are reported in Table VIII. The results show that our method requires 61.85 times less CPU time than tool TimberWolf. Additionally, the HPWL is competitive to obtain an improvement in wirelength over TimberWolf for all the five circuits. On average, our method yields the wirelength that is 21.79% shorter than this simulated annealing based commercial tool.
TABLE VII
RESULTS REPORTED BY TimberWolf

<table>
<thead>
<tr>
<th>MCNC Circuit</th>
<th>Total Area (mm²)</th>
<th>TimberWolf [37]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Total Wirelength (mm)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Runtime (s)</td>
</tr>
<tr>
<td>apte</td>
<td>56.00</td>
<td>66.00</td>
</tr>
<tr>
<td>xerox</td>
<td>22.64</td>
<td>101.20</td>
</tr>
<tr>
<td>hp</td>
<td>8.30</td>
<td>91.40</td>
</tr>
<tr>
<td>ami33</td>
<td>1.16</td>
<td>221.00</td>
</tr>
<tr>
<td>ami49</td>
<td>35.4</td>
<td>472.80</td>
</tr>
</tbody>
</table>

TABLE VIII
IMPROVEMENTS IN RUNTIME AND WIRE LENGTH COMPARED WITH TimberWolf

<table>
<thead>
<tr>
<th>MCNC Circuit</th>
<th>Speed-up</th>
<th>WL Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>apte</td>
<td>64.08</td>
<td>+18.60%</td>
</tr>
<tr>
<td>xerox</td>
<td>90.36</td>
<td>+22.00%</td>
</tr>
<tr>
<td>hp</td>
<td>70.85</td>
<td>+23.97%</td>
</tr>
<tr>
<td>ami33</td>
<td>36.47</td>
<td>+30.25%</td>
</tr>
<tr>
<td>ami49</td>
<td>47.47</td>
<td>+14.15%</td>
</tr>
<tr>
<td>Average</td>
<td>61.85</td>
<td>+21.79%</td>
</tr>
</tbody>
</table>

E. Obtaining Zero-Deadspace Floorplans

The results reported above show that our methodology is constantly competitive, and frequently outperforms, the methods of Murata and Kuh, Adya and Markov, and TimberWolf. Beyond these features, a very important feature of our methodology is that not only do the dimensions of the floorplans in our experiments comply with the original ones provided in the MCNC benchmark, but also zero-deadspace floorplans were obtained for all five circuits, as shown in Table IX. Fig. 3D, Fig. 5B and Fig. 10, respectively, depict the best zero-deadspace floorplans for the benchmarks apte, ami33 and ami49 obtained using our methodology.

Thus, our approach is able to guarantee complete area utilization in a fixed-outline situation, while MK, AM and TimberWolf have some deadspace in their floorplans.

F. Comparison with Parquet, Capo, IMF and IMFAFF models

For large circuits of the GSRC benchmark, we compare our model with Parquet [1], [35], Capo [4], IMF and IMFAFF [5], where I/O pads were shifted to the boundary of the chips. In our experiment, I/O pads also were fixed on the boundary of the chips for comparison. We use the three largest circuits n100, n200, and n300 from the GSRC benchmarks. All reported wirelengths were measured using the HPWL, which was also used by Parquet, Capo, IMF and IMFAFF. Each circuit was run once and the floorplans obtained are reported in Table X. For all the circuits, our model can achieve complete area utilization, i.e., zero-deadspace, in a fixed-outline. The results of Parquet, Capo, IMF and IMFAFF described in [5] only stated that the whitespace allowed is less than 15%. Therefore, we chose our case of 10% whitespace to compare with these floorplanners for HPWL wirelength in Table XI. Our model can obtain better wirelength solutions than Parquet, Capo, IMF and IMFAFF, except IMF for n300a circuit. Our model gives an improvement of 0.33% to 23.88% in terms of the total wirelength in comparison with these floorplanners. Parquet and Capo can handle hard modules and soft modules. Our methodology is currently focused on the soft modules but in principle it is applicable to the hard module case. We are uncertain which kind of modules IMF and IMFAFF used.

G. Comparison with Parquet, and CC models

Next, our model is compared with the Chen and Chang model (CC) [6], and Parquet [1], [35], with the I/O pads fixed at the locations originally given by the benchmarks. In fact, in this case, the experimental results of Parquet were also reported in [6] for comparison. In our experiments, the HPWL wirelength computation method and the model itself are the same as above, but I/O pads were fixed at the original locations in the benchmarks for comparisons.

Our model allows the flexibility to have any amount of whitespace. Not only can our model implement zero-deadspace floorplanning, but also it implements any percentage of whitespace, by setting the whitespace fraction $\gamma$ to the desired value. Our test results for zero-deadspace and 5% whitespace are reported in Table XII, and the comparisons with CC model and Parquet for 10% and 15% whitespace, are listed in Table XIII and Table XIV, respectively. The data of zero-deadspace and 5% whitespace cases for CC model and Parquet are not available in their papers. The results reported in their papers were obtained by defining the maximum percentage of whitespace as 10% and 15%. However, the specific percentage of whitespace they adopted was not pointed out in the papers. Therefore, we compare our experimental results under both percentages of whitespace, 10% and 15%, with theirs. Table XIII shows that our model achieves improvements of 9.55%, 11.65%, and 15.79% in the total wirelength compared to CC model for n100, n200, and n300, respectively, with 10% whitespace. In comparison with Parquet, our model further improves the total wirelength by 13.58%, 18.91%, and 21.38% for n100, n200, and n300, respectively, with 10% whitespace. It is a superior feature that our model can produces greater improvement in terms of the total wirelength with increasing number of modules compared with the other two floorplanners (see Table XIII and Table XIV). Particularly compared with Parquet, the total wirelength is reduced by 13.58% to 21.38% from n100 to n300, respectively.

Our methodology is also compared with the model of Zhan et al. [13] (in short, ZPS). The floorplans with 15% whitespace obtained by Zhan et al.'s model and Parquet4 dealing with soft modules are reported in [13]. Table XV lists the comparison result when I/O pads are fixed at the original locations in the benchmarks. Table XV again shows that for soft modules, we can produce greater total wirelength improvement as the number of modules increases compared with these two floorplanners.

A very important feature of our methodology is that not only do the dimensions of the floorplans in our experiments comply with the original ones provided in the GSRC benchmark, but also zero-deadspace floorplans can be obtained. Thus, our
approach is able to guarantee complete area utilization in a fixed-outline situation.

**VIII. CONCLUSION**

We proposed a two-stage completely convex optimization methodology for floorplanning that can be applied to both classical floorplanning and fixed-outline floorplanning. Computational results on the GSRC benchmark demonstrate that our methodology clearly outperforms the results reported in the literature. A very important feature of our methodology is that not only do the dimensions of the floorplans in our experiments comply with the original ones provided in the

---

**TABLE IX**

<table>
<thead>
<tr>
<th>MCNC Circuit</th>
<th>Total Area (mm²)</th>
<th>MK [17] Area (mm²)</th>
<th>Area Deadspace</th>
<th>AM [1] Area min/avg (mm²)</th>
<th>Deadspace</th>
<th>TimberWolf [36], [37] Area (mm²)</th>
<th>Deadspace</th>
<th>Our Methodology Area (mm²)</th>
<th>Deadspace</th>
</tr>
</thead>
<tbody>
<tr>
<td>apte</td>
<td>46.56</td>
<td>46.55</td>
<td>-0.02%</td>
<td>46.97/48.95</td>
<td>0.87/4.88%</td>
<td>48.50</td>
<td>4.00%</td>
<td>46.56</td>
<td>0%</td>
</tr>
<tr>
<td>xerox</td>
<td>19.35</td>
<td>19.50</td>
<td>0.77%</td>
<td>19.51/20.62</td>
<td>0.82/6.16%</td>
<td>22.64</td>
<td>14.53%</td>
<td>19.35</td>
<td>0%</td>
</tr>
<tr>
<td>hp</td>
<td>8.30</td>
<td>8.83</td>
<td>6.0%</td>
<td>8.96/9.72</td>
<td>7.40/14.60%</td>
<td>9.58</td>
<td>13.36%</td>
<td>8.30</td>
<td>0%</td>
</tr>
<tr>
<td>ami33</td>
<td>1.16</td>
<td>1.16</td>
<td>0%</td>
<td>1.18/1.24</td>
<td>1.70/6.45%</td>
<td>1.27</td>
<td>8.66%</td>
<td>1.16</td>
<td>0%</td>
</tr>
<tr>
<td>ami49</td>
<td>35.4</td>
<td>35.58</td>
<td>0.5%</td>
<td>36.07/37.8</td>
<td>1.86/6.35%</td>
<td>40.81</td>
<td>13.26%</td>
<td>35.4</td>
<td>0%</td>
</tr>
</tbody>
</table>

**TABLE X**

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Zero-whitespace</th>
<th>5% whitespace</th>
<th>10% whitespace</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU time(s)</td>
<td>HPWL</td>
<td>CPU time(s)</td>
</tr>
<tr>
<td>n100</td>
<td>55.89</td>
<td>197490</td>
<td>55.32</td>
</tr>
<tr>
<td>n200</td>
<td>1026.43</td>
<td>356520</td>
<td>1129.14</td>
</tr>
<tr>
<td>n300</td>
<td>1654.70</td>
<td>477800</td>
<td>1669.59</td>
</tr>
</tbody>
</table>

**TABLE XI**

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Our</th>
<th>Parquet</th>
<th>Capo</th>
<th>IMF</th>
<th>IMFAFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPWL</td>
<td>HPWL</td>
<td>HPWL</td>
<td>HPWL</td>
<td>HPWL</td>
<td>HPWL</td>
</tr>
<tr>
<td>n100</td>
<td>203700</td>
<td>242050</td>
<td>234390</td>
<td>207852</td>
<td>208772</td>
</tr>
<tr>
<td>n200</td>
<td>367880</td>
<td>432882</td>
<td>385594</td>
<td>369888</td>
<td>372845</td>
</tr>
<tr>
<td>n300</td>
<td>492830</td>
<td>647452</td>
<td>522968</td>
<td>489868</td>
<td>494480</td>
</tr>
</tbody>
</table>
GSRC benchmark, but also zero-deadspace floorplans can be obtained. Thus, our approach is able to guarantee complete area utilization in a fixed-outline situation.

Our methodology guarantees complete area utilization in a fixed-outline situation, and it also produces floorplans with any specified percentage of whitespace. Most importantly, our methodology provides greater improvement over other floorplanners as the number of modules increases.

REFERENCES


