Floor Planning Problem– A Convex Optimization Approach

Mingju Liu
ECE Department
Rutgers University
Piscataway, NJ
mingju.liu@rutgers.edu

***Abstract*—The Floor Planning problem can be categorized as a combinatorial optimization problem where the goal is to determine the optimal position and dimensions of a set of geometric shapes within a space, such that there is no overlap between. Two approaches are proposed to solve this problem. As the direct objective function is not convex, the main process is to reformat the problem as a convex problem. One is provided by Boyd Textbook and using Linear Programming, the other is to form the problem as a Geometric Programming problem. After implementing these two approaches, I can acquire the optimal floor planning as the problem described.**

***Index Terms*—floor planning, convex optimization, linear programming, geometric programming**

I. Introduction

Floor planning problem is a classic problem in optimization field. The goal is to determine the optimal position and dimensions of a set of geometric shapes within a space, such that there is no overlap between. We are going to look at the problem of placing axis-aligned rectangles inside a 2D plane, or more specifically, inside a bounding rectangle. We will also explore various geometric constraints that can be applied.

This problem has many applications in fields such as architecture (Fig. 1) and VLSI design (Fig. 2). In these two fields, there are also different objectives.

In architecture, the rectangles can correspond to rooms on a floor. The room can be restrained to having a minimum or maximum area and the walls can be aligned with each other in various ways. The area of the floor is usually fixed, so the objective would be to arrange rooms so that they cover the whole area of the floor while make sure the constraints are satisfied.

In VLSI design, during the Physical design cycle, the millions of logic components that build up a circuit are partitioned into blocks. The blocks are interconnected by signals and have a minimum area defined by the components from which it is built up. The goal is to place the blocks so that the area of the chip is minimized and that certain placement constraints are respected.

Therefore, in general, the objectives in floor planning problem can be either (1) to maximize the area of all blocks while placing them inside a bounding rectangle of fixed area or (2) to determine the dimensions and placement of blocks that minimize the area of the bounding rectangle.

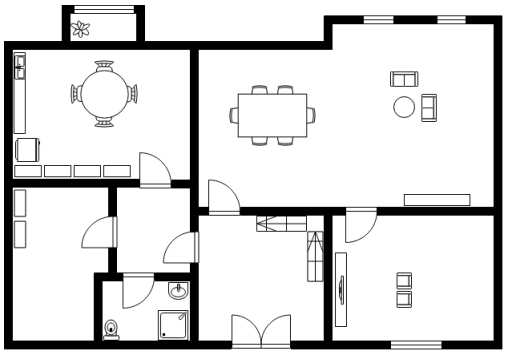

Fig. 1. Floor Planning in Architecture

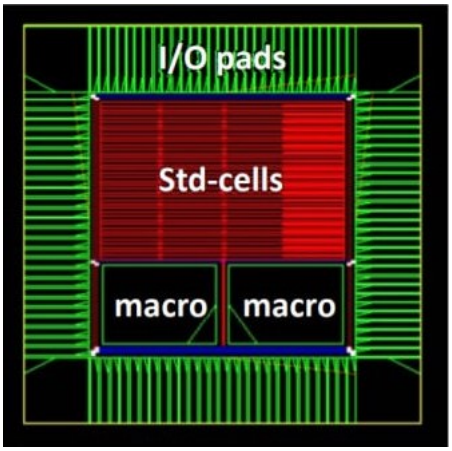

Fig. 2. Floor Planning in VLSI design

II. Related Work

A. Two-stage optimization methodology

In this paper, a two-stage optimization methodology is proposed to solve the fixed-outlined floorplanning problem that is a global optimization problem for wirelength minimization. In the first stage, a 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. So that overlap-free and deadspace-free floorplans are achieved in a fixed-outline and floorplans with any specified percentage of whitespace can be produced.

I implemented the first-stage at the beginning and tried to produce the relative positions using the provided methodology – Relative Position Matrix (RPM). And then I learn from the second stage for the aspect ratio constraints to solve this problem in my next form of this question.

B. Floor planning in Convex Optimization Textbook

In the convex optimization Textbook, the author First introduced the placement problems and the floor planning problem can be considered an extension of it. Then the author solved the problem in the convex optimization form and using linear function and linear constraints. I first implemented all the parts introduced in the Textbook and attain a high-accuracy result in the end.

C. Geometric programming approach for Floor Planning problem

In this paper, the author provides another method to solve the floorplanning problem. To solve this problem in geometric programming form, we need to derivate the objective function and also constraints to be in geometric form. After I implemented this methodology using GP solver, I also do the perturbation analysis to attain better conclusions.

III. Problem Statement

In this section, we will define the Floor Planning Problem as a general optimization problem.

We have N blocks denoted as B_1, ...,B_N that are configured and placed inside a bounding rectangle with width W and height H, as shown in Fig. 3. Each block Bi is defined by the coordinates of its lower left corner (x_i, y_i), its width w_i and its height h_i. Coordinates (0,0) correspond to the lower left corner of the bounding rectangle which will be utilized to general the ultimate layout diagram.

To simplify this problem, we will only consider the floor planning problem in VLSI design field. That means, our objective is to determine the dimensions and placement of the provided blocks that minimize the area of the bounding rectangle, which the objective function will be:

$$minimize\ WH \tag{1}$$

We can obtain some constraints from the previous description. The blocks must lie within the bounding box (or rectangle), i.e.

$$\begin{aligned} &x_i \geq 0, \\ &y_i \geq 0, \\ &x_i + w_i \leq W, \\ &y_i + h_i \leq H, i = 1, ..., N \end{aligned} \tag{2}$$

The blocks cannot overlap with each other, i.e.

$$int(B_i \cap B_j) = \varnothing, i \neq j \tag{3}$$

In general, the variables of this problem are x_i, y_i, w_i, h_i for i = 1, …, N, as well as W and H. All of the variables need to be positive real numbers. The area of the i-th block will be $a_i = h_i w_i$. It can be defined to be fixed at a real positive number or floating within limits. In this report, we will define the blocks have fixed values, but the aspect ratio of the width and height $\alpha = h_i / w_i$ will be changeable which is also a variable in this problem.

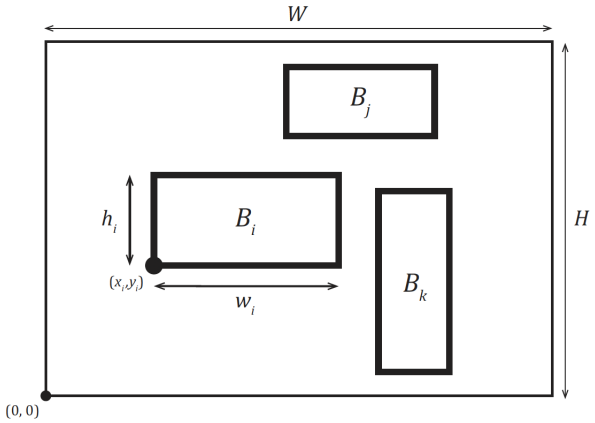

Fig. 3. The Floorplanning Problem

While trying to solve this problem, we will find this is a NP-hard problem. We can use convex optimization approach to be denoted as a solution. However, the objective function as shown in (1) is not a convex function if W and H are both variables. In this way, we need to do convex relaxation in the next process. I will implement two methods to reconstruct it as a convex optimization problem in the next section.

IV. Convex Formulation

In order to pose the floor planning problem as mentioned before as a convex problem, we need to do the convex relaxation and formulate it so that the objective function and the inequality constraints are convex functions, and that the equality constraints are affine functions.

As mentioned, this problem can be formulated in two different ways.

A. Linear Programming

1) Objective Function:

According to the definition of convex function, the area of the bounding box A = WH is not a convex function in both W and H because its Hessian has eigenvalues -1 and 1. So we need to take the logarithm of this area, and we can get logA = logW + logH which is a concave function because sum of two concave functions is concave function. However, we can only maximize the concave function, not minimize it.

We can minimize the opposite value of logA, but because the blocks and the bounding box are both rectangles, we can minimize the perimeter of the bounding rectangle which is an affine function.

$$minimize\ 2(W + H) \tag{4}$$

Since a rectangle with minimal area can turn out to be very long or narrow, the minimal perimeter objective would be

better because it incorporates the tendency to form a bounding box with equal slides, because we know with equal area, perimeter of the square will be the smallest. A square chip will also preferential in VLSI design.

2) Constraints:

In this section we will first consider the basic constraints discussed before, and then add some constraints we may use in special cases.

- Bounding rectangle constraints:
  These constraints are defined in (2). All of these four constraints are affine functions, and their intersection is a convex set (a rectangle), we can utilize these constraints directly.
- Overlap free constraints:
  The constraint from (3) cannot be utilized in the convex optimization problem directly obviously. Then we need to find an expression of this constraint. As shown in Fig. 4, this is an example of B_i and B_j, so the boundary of the feasible set is a rectangle obtained by slide one block around the other. With B_j sliding around B_i, the dimensions of the shaded rectangle are $(w_i + w_j)(h_i + h_j)$. Also if we define the lower left corner is the coordinate of position, we will get its position is $(x_i - w_j, y_i - h_j)$.

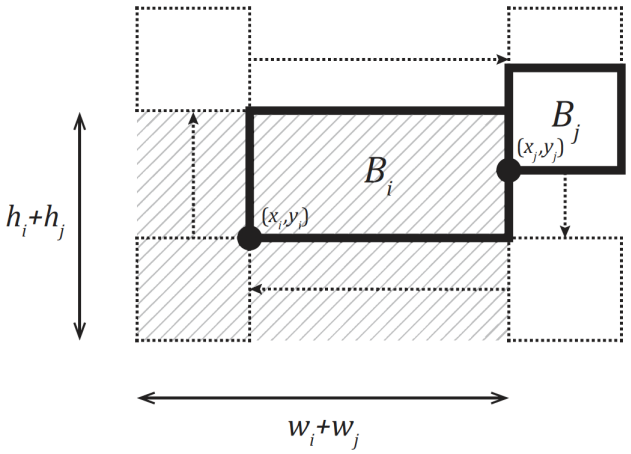

Fig. 4. Overlap free constraints

Since the feasible set defined by this constraint represents the whole space, except the interior of the shaded rectangle, it is clear graphically that this constraint is not convex. However, we can impose some limitations on this overlap-free constraint in order to treat it as a convex optimization problem.

We can tell from the Fig. 4 that this feasible set is a union of 4 half-spaces, and half spaces is convex, we can utilize one of them to pose the constraints. That is equivalent to choosing a relative positioning between blocks. In other words, for any two blocks B_i and B_j ($i \neq j$) hold one of the following four relations or constraints.

$$\begin{aligned} &B_i \ is\ to\ the\ left\ of\ B_j \quad x_i + w_i \leq x_j \\ &B_i \ is\ to\ the\ right\ of\ B_j \quad x_j + w_j \leq x_i \\ &B_i \ under\ B_j \quad y_i + h_i \leq y_j \\ &B_i \ above\ B_j \quad y_j + h_j \leq y_i \end{aligned} \tag{5}$$

By imposing relative positioning between blocks, there will be N(N-1) / 2 affine inequality constraints. Also there will be transitivity of the relative positions. For instance, if B_i is above B_j, and B_j is above B_k, then B_i will also be above B_k. From this, we can validation this relative position constraints.

In order to specify the relative position, we will introduce the following two binary relations on the set of indices $\{1, \ldots, N\}$: L ('left of') and U ('under'). So the right and above relative positions will be the symmetric relations. Then we can impose the constraints that block B_i is to the left of block B_j if (i, j) $\in L$, and block B_i is under block B_j if (i, j) $\in U$, i.e:

$$\begin{aligned} x_i + w_i \leq x_j \iff (i,j) \in L \\ y_i + h_i \leq y_j \iff (i,j) \in U \end{aligned} \tag{6}$$

To solve the Floorplanning problem, we need to provide as an input of the two relations L and U, which will be the valid relative positions between each pair of blocks. The input will be in the matrix format, which will be the Relative Position Matrix (RPM) [1].

- Block Aspect Ratio:
  We can introduce another constraint to ensure the blocks will not be too long or narrow which is the aspect ratio of the blocks. We will impose upper and lower bounds to the aspect ratio of each cell:

$$r_i^{min} \leq h_i / w_i \leq r_i^{max} \tag{7}$$

For this linear programming approach, we can reconstruct it as a linear expression which can be used for equality constraints for a fixed value of the aspect ratio:

$$w_i r_i^{min} \leq h_i \leq w_i r_i^{max} \tag{8}$$

- Primal Convex Problem:
  In previous sections we have discussed many variations of objectives and constraints, we will derive the basic version of this problem: Minimize the area of the bounding box for a set of N rectangular blocks, where no blocks overlap with each other and each block B_i has a minimum area A_i.
  Then the variables will be:
  1) Dimensions of the bounding rectangle: W, H $\in \mathbb{R}_+$
  2) Four vectors representing x and y coordinates of blocks, widths and heights: x, y, w, h $\in \mathbb{R}_+^N$

  Then the input data will be:
  1) Minimal areas of all blocks: $A_i^m in$, i = 1, ..., N
  2) Relative position matrix of L and U.

We can reorganize this to be the standard form:

$$minimize\ 2(W+H) \tag{9}$$

$$\begin{aligned} subject\ to\ & -x \preceq 0, -y \preceq 0, -w \preceq 0, -h \preceq 0 \\ & -W \leq 0, -H \leq 0 \\ & x + w - W \cdot \vec{1} \preceq 0, y + h - H \cdot \vec{1} \preceq 0 \\ & A_i/h_i - w_i \leq 0, i = 1, ..., N \\ & L_{ij} \cdot (x_i + w_i - xj) \leq 0 \\ & U_{ij} \cdot (y_i + h_i - yj) \leq 0 \end{aligned} \tag{10}$$

B. Geometric Programming

A Geometric Program (GP) is a type of mathematical optimization problem characterized by objective and constraint functions that have a special form. For this Floor Planning Problem, we can tell the objective functions are nonconvex and some of the constraints either. So we can derive this problem as a GP problem and then solve.

It's useful to compare GP modeling and modeling via general purpose nonlinear optimization (NLP). Because objective functions and constraints of NLP can be any nonlinear function, GP can be much tricker. But solving GP is much easier than NLP. In return for accepting this limitation, though, we get the benefit of extremely efficient and reliable solution methods, that scale gracefully to large-scale problems [3].

1) Standard GP modeling approach:
The Geometric Program has a standard form as following,

$$minimize\ f_0(x) \tag{11}$$

$$\begin{aligned} subject\ to\ & f_i(x) \leq 1, i = 1, ..., m \\ & g_i(x) = 1, i = 1, ..., p \end{aligned} \tag{12}$$

where f_i are posynomial, g_i are monomials, and x_i are the optimization variables. The variables have to be positive (x_i ¿ 0). In standard form, the objective function should be posynomial which has to be minimized, equality constraint should be monomials equal to one and inequality constraint should have posynomails less than or equal to one.

The standard monomials and posynomials are defined as below:

- Monomial

$$f(x) = cx_1^{a_1}x_2^{a_2}...x_n^{a_n}\ \ c > 0, a_i \in \mathbb{R} \tag{13}$$

- Posynomial: sum of monomials

$$f(x) = \sum_{k=1}^{K} c_k x_1^{a_{1k}} x_2^{a_{2k}} ... x_n^{a_{nk}}\ \ c_k > 0, a_{ik} \in \mathbb{R} \tag{14}$$

where, x_1, ..., x_n denote n real positive variables.

2) Generalized Geometric Programming:
As the standard geometric program requires standard form of posynomial and monomial, there are still some format may look like the standard form but not standard. They are called generalized posynomials and can be categorized in the Generalized Geometric Programming (GGP). In this section, we will discuss different types of posynomials.

- Fractional powers of posynomials:
  If the constraints involve fraction powers, such as $f(x)^2.2$, it's not a standard posynomial but a generalized posynomial. They can still be solved via GP approach by introducing a new variable and a bounding constraint.
- Maximum of posynomials:
  We can tell that any maximum of posynomials is not posynomial as it is generally not differentiable, whereas a posynomial is everywhere differentiable. By introducing new variable and new inequalities, the maximum can also be inverted to be a GP problem.
- Generalized Posynomials in general:
  We can say that a function f of positive variables x_1, ..., x_n is a generalized posynomial if it can be formed from posynomials using operations of addition, multiplication, positive (fractional) power, maximum as well as other operations that can be derived from these, such as division by monomials. They are also closed under composition in the following sense. If f_0 is a generalized posynomial of k variables, for which no variable occurs with a negative exponent, and f_1, ..., f_k are generalized posynomials, then the composition function

$$f_0(f_1(x), ..., f_k(x)) \tag{15}$$

  is a generalized posynomial.

Thus we can give the Generalized Geometric Programming as follows:

$$minimize\ f_0(x) \tag{16}$$

$$\begin{aligned} subject\ to\ & f_i(x) \leq 1, i = 1, ..., m \\ & g_i(x) = 1, i = 1, ..., p \end{aligned} \tag{17}$$

where g_1, ..., g_p are monomials and f_0, ..., f_m are generalized posymials as introduced in[15]. Note that, since any posynomial is also a generalized posynomial, any GP is also a GGP.

3) Floor Planning Problem in GGP:
As mentioned in previous sections, the constraint that the rectangles do not overlap makes the general floor planning problem a complicated combinatorial optimization problem. However, if the relative positioning of the boxes is specified, this can be formulated as a GP problem. In this section, we will discuss the formulation. We will consider a specific example with 4 blocks, labeled A, B, C, D, as shown in Fig. 5. And the relative positioning will be as following:

- A is to the left of B.
- C is to the left of D.
- A and B are above C and D.

This will guarantee the overlap free constraints.

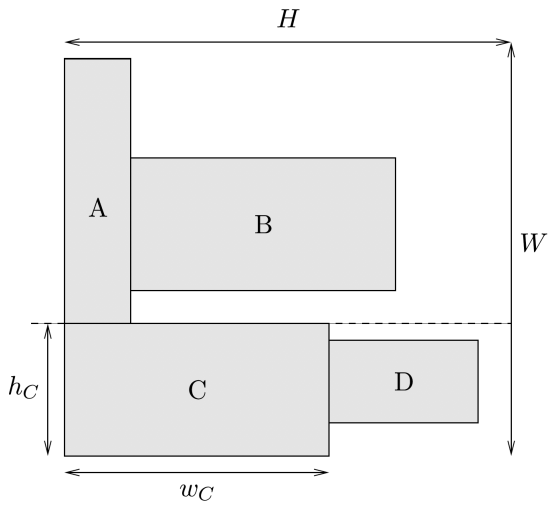

Fig. 5. GP based Floor Planning Example

The width of A and B together is w_A + w_B, and the width of C and D together is w_C + w_D. Therefore the width of the bounding box is

$$W = max\{w_A + w_B, w_C + w_D\} \tag{18}$$

The height of A and B together is max $\{h_A, h_B\}$, and the height of C and D together is max$\{h_C, h_D\}$, so the height of the bounding box is

$$H = max\{h_A, h_B\} + max\{h_C, h_D\} \tag{19}$$

The area of the bounding box is

$$\begin{aligned} WH =& max\{w_A + w_B, w_C + w_D\} \cdot \\ & (max\{h_A, h_B\} + max\{h_C, h_D\}) \end{aligned} \tag{20}$$

As (15) mentioned, the area can be recognized as a generalized posynomial of the varirabels w_A, ..., w_D and h_A, ..., h_D. As for the constraints, we will first input the area of the four blocks. To utilize the constraints to make the bounding box more preferential, we will use the aspect ratio as introduced in(7, and we introduce a new variable α as the aspect ratio. Thus we can obtain the Optimizaiton problem

$$minimize\ WH \tag{21}$$

$$\begin{aligned} subject\ to \quad & h_A w_A = a \\ & h_B w_B = b \\ & h_C w_C = c \\ & h_D w_D = d \\ & 1/\alpha_{max} \le h_A/wA \le \alpha_{max}, ..., \\ & 1/\alpha_{max} \le h_D/wD \le \alpha_{max} \end{aligned} \tag{22}$$

This is a GGP, with variables w_A, ..., w_D and h_A, ..., h_D. a, b, c, d and α_max are the input values. After formulation, the floorplanning problem can be solved in GGP approach.

V. Convex Solver

A. Linear Programming

To solve the Linear Programming Problem we generated in (9) (10), we need to utilize the LP solver. First, I generated a function named optimalPlacement to solve the convex optimization problem in general. In the Test.m file, I input the Relative positioning Matrix and the block size manually, in this context, my input is

$$L = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{23}$$

$$U = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{24}$$

And the block size is 30, 150, 30, 200 respectively. With solved x, y, w, h, W, H values, I can draw the layout of the optimal floorplan.

B. Geometric Programming

In this section, we will try to use GPsolver to implement the generated problem in (21) (22). We choose ggplab as our solver. **ggplab** is a Matlab-based toolbox for specifying and solving geometric programs (GPs) and generalized geometric programs (GGPs). As our problem is a GGP, it can be solved by **ggplab**.

ggplab consists of:

- gpcvx, a primal-dual interior-point solver for GP (in convex form) and a wrapper, gpposy, that accepts GPs given in posynomial form.
- A library of objects, such as monomials, posynomials, and generalized posynomials, to support the specification of GPs and GGPs.

Then I input the block size and also the aspect ratio manually. The block size I choose is the same as the LP. Aspect Ratio I choose is α = 2 and 4. After the solver elapsed the problem successfully. I will also draw the layout of the optimal result to make it more visualized.

VI. Results

A. Results of LP

After input the block sizes and Relative Positioning Matrix, we can get the result as Fig. 6 shown. This diagram has 4 results related to 4 floor plans. Note that, the sequence number 1, 2, 3, and 4 are the fixed block number. Whereas their block size is not fixed. My input of 30, 150, 30, 200 will be randomly distributed to the four blocks. The left two results of them is more preferential according to our constraints. The lower right result has way too narrow block 1, and this may not be accepted in real VLSI design. The Floor Plan we got are all square bounding box, these are due to the objective function (4). Then the width and height of the square will be same at about 20.48, and the whole area will be 410 obviously which is the sum of all four blocks.

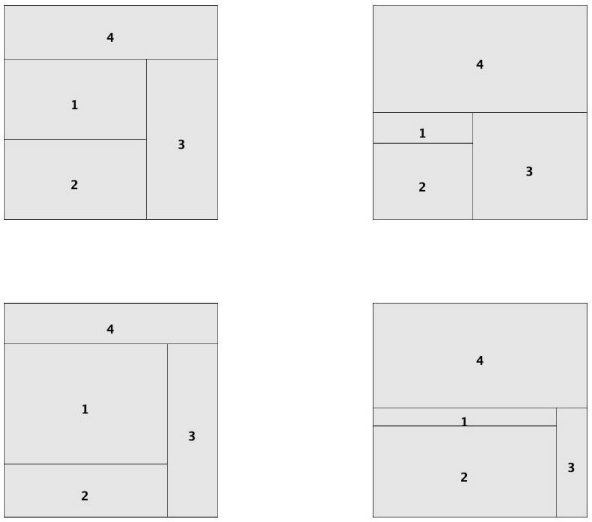

Fig. 6. Floor plans after Linear Programming approach. The relative positioning follows the RPM we input.

B. Results of GP

After I input the block sizes and the value of α, I can get the result from the ggplab solver. Here are the results of α = 2 and α = 4 respectively.

Fig. 7. Floor Plans after GP solver. α = 2

In Fig. 7, we can see there is still unused space in the bounding box obviously. The result from the GP solver, is that the area of the whole bounding box is about 428 which is larger than the sum of the 4 block sizes. In this way, we can know that the aspect ratio α = 2 is not a proper value in this GGP problem. Note that, in this diagram, I use four colors to represent 4 blocks and the relative positioning is the same as Fig. 5 shown in the previous section.

In Fig. 8, we can tell this is the optimal floor plan with no unused space inside the bounding box. From the results of the GP solver, the area of the whole bounding box is 410, which is exactly the sum of the four block sizes and the same as the result from the LP solver. Note that, the colors I used to represent the four blocks are the same as the first result we obtain from the previous part of GP. And we can argue α = 4 is a feasible value for this GGP approach.

Fig. 8. Floor Plans after GP solver. α = 4

C. Result in general

From the two results provided in the previous section, we can tell some differences between them. Due to the settings of the objective function in the LP approach, the floor plans will be squares, the aspect ratio of the blocks placed in the bounding box will be the trade off. Some of the floor plans will be too long or narrow. As in the GP approach, the choice of the proper aspect ratio will be relatively important. If we choose some value imporporiate, there will be some unused space in the result. When choosing a more proper value of aspect ratio, the unused space will be vanished. Note that, the bounding box will be regular rectangle after the GGP approach, because the objective function is minimizing the area directly not the perimeter. As trading off, the aspect ratio of the blocks will be more preferential which means there will be no too long or narrow blocks. We will analysis which value of the aspect ratio α will be optimal or how will the result change while α changing in the next section.

VII. Analysis

A. Lagrangian Function and Dual Form

We will derive the dual form of the LP approach. For deriving the dual form, we need to write the Lagrangian with the following Lagrangian parameters:

- Vectors λ_1, λ_2, λ_3, λ_4, λ_7, λ_8, $\lambda_9 \in \mathbb{R}_+^N$ for the four boundary constraints, for the constraining w and h to positive numbers and for the minimum area constraint in (10)
- Scalars λ_5 and λ_6 for cosntraining W and H to positive numbers
- Matrices Λ_{10}, $\Lambda_{11} \in \mathbb{R}_+^{N\times N}$ for the horizontal and vertical no-overlap constraints.

Here is the Lagrangian function L(W, H, x, y, w, h, λ_1, λ_2, λ_3, λ_4, λ_5 and λ_6, λ_7, λ_8, λ_9, Λ_{10}, Λ_{11})

$$\begin{aligned}L(\cdots) =&2(W+H)+\sum_{i=1}^{N}\lambda_{1i}(-x_i)+\sum_{i=1}^{N}\lambda_{2i}(-y_i)\\&+\sum_{i=1}^{N}\lambda_{3i}(-w_i)+\sum_{i=1}^{N}\lambda_{4i}(-h_i)-\lambda_5 W-\lambda_6 H\\&+\sum_{i=1}^{N}\lambda_{7i}(x_i+w_i-W)+\sum_{i=1}^{N}\lambda_{8i}(y_i+h_i-H)\\&+\sum_{i=1}^{N}\lambda_{9i}(\frac{A_i}{h_i}-w_i)\\&+\sum_{i=1}^{N}\sum_{j=1}^{N}\Lambda_{10ij}\cdot L_{ij}\cdot(x_i+w_i-x_j)\\&+\sum_{i=1}^{N}\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij}\cdot(y_i+h_i-y_j)\end{aligned} \tag{25}$$

Then,

$$\begin{aligned}L(\cdots) =&W(2-\lambda 5-\sum_{i=1}^{N}\lambda_{7i})+H(2-\lambda 6-\sum_{i=1}^{N}\lambda_{8i})\\&+\sum_{i=1}^{N}x_i(-\lambda_{1i}+\lambda_{7i}+\sum_{j=1}^{N}\Lambda_{10ij}\cdot L_{ij})\\&+\sum_{i=1}^{N}y_i(-\lambda_{2i}+\lambda_{8i}+\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij})\\&-\sum_{i=1}^{N}\sum_{j=1}^{N}\Lambda_{10ij}\cdot L_{ij}\cdot x_j-\sum_{i=1}^{N}\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij}\cdot y_j\\&+\sum_{i=1}^{N}w_i(\lambda_{7i}-\lambda_{3i}-\lambda_{9i}+\sum_{j=1}^{N}\Lambda_{10ij}\cdot L_{ij})\\&+\sum_{i=1}^{N}(h_i(\lambda_{8i}-\lambda_{4i}-\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij})+\frac{\lambda_{9i}A_i}{h_i})\end{aligned} \tag{26}$$

Then we will make the following transformation to the 5th term of (26):

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\Lambda_{10ij}\cdot L_{ij}\cdot x_j \;=\; \sum_{i=1}^{N}x_j\cdot\sum_{j=1}^{N}\Lambda_{10ij}\cdot L_{ij} \tag{27}$$

and also do the same for 6th term, additionally swapping i and j in those two terms, we could get:

$$\begin{aligned}L(\cdots) =&W(2-\lambda 5-\sum_{i=1}^{N}\lambda_{7i})+H(2-\lambda 6-\sum_{i=1}^{N}\lambda_{8i})\\&+\sum_{i=1}^{N}x_i(-\lambda_{1i}+\lambda_{7i}+\sum_{j=1}^{N}\Lambda_{10ij}\cdot L_{ij}-\sum_{j=1}^{N}\Lambda_{6ji}\cdot L_{ji})\\&+\sum_{i=1}^{N}y_i(-\lambda_{2i}+\lambda_{8i}+\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij}-\sum_{j=1}^{N}\Lambda_{7ji}\cdot U_{ji})\\&+\sum_{i=1}^{N}w_i(\lambda_{7i}-\lambda_{3i}-\lambda_{9i}+\sum_{j=1}^{N}\Lambda_{10ij}\cdot L_{ij})\\&+\sum_{i=1}^{N}(h_i(\lambda_{8i}-\lambda_{4i}-\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij})+\frac{\lambda_{9i}A_i}{h_i})\end{aligned} \tag{28}$$

To derive the dual function, we need to find the infimum of the Lagrangian over all decision variables:

$$\begin{aligned}&g(\lambda_1,\lambda_2,\lambda_3,\lambda_4,\lambda_5 and\lambda_6,\lambda_7,\lambda_8,\lambda_9,\Lambda_{10},\Lambda_{11})\\&=\inf_{W,H,x,y,w,h}L(W,H,x,y,w,h,\lambda_1,\lambda_2,\lambda_7,\lambda_8,\lambda_9,\Lambda_{10},\Lambda_{11})\end{aligned} \tag{29}$$

We can see in (28) that all factors, except the last one, are linear in their respective variables. Therefore, the terms in brackets next to W, H, x, y and w must be set to zero, otherwise the whole infimum becomes unbounded below and, as a result, the value of the dual function becomes -∞. As for the last term, since it's non-linear, we need to find the value of h that minimizes it. We do that by finding the gradient and setting it to zero. We will do it for every h_i:

$$h_i^*=\sqrt{\frac{\lambda_{9i}A_i}{\lambda_{8i}-\lambda_{4i}+\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij}}} \tag{30}$$

We will replace h_i in the original term with h_i^*, we can write our dual function:

$$\begin{cases}2\sum_{i=1}^{N}\sqrt{\lambda_{9i}A_i(\lambda_{8i}-\lambda_{4i}+\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij})},\\ h_1^{'}=h_2^{'}=h_3^{'}=h_4^{'}=h_5^{'}=0\\ -\infty, \qquad otherwise\end{cases} \tag{31}$$

And we can formulate the dual problem:

$$maximize\ 2\sum_{i=1}^{N}\sqrt{\lambda_{9i}A_i(\lambda_{8i}-\lambda_{4i}+\sum_{j=1}^{N}\Lambda_{11ij}\cdot U_{ij})} \tag{32}$$

$$\begin{aligned}subject\ to\ &2-\vec{1}^T\lambda_7=0\\&2-\vec{1}_T\lambda_8=0\\&\lambda_1-\lambda_7+(diag(\Lambda_{10}L^T)-diag(\Lambda_{10}^TL))\cdot\vec{1}=0\\&\lambda_8-\lambda_2+(diag(\Lambda_{11}U^T)-diag(\Lambda_{11}^TU))\cdot\vec{1}=0\\&\lambda_7-\lambda_9+diag(\Lambda_{10}L^T)\cdot\vec{1}=0\\&\lambda_1\preceq 0,\lambda_2\preceq 0,\lambda_7\preceq 0,\lambda_8\preceq 0,\lambda_9\preceq 0\\&\Lambda_{10ij}\preceq 0,\Lambda_{11ij}\preceq 0,i=1,...,N,j=1,...,N\end{aligned} \tag{33}$$

B. Pertubation Analysis in GP

We will do the perturbation analysis on our GP approach. In the GP constraints, we can note that the controlled or perturbed value we can set is the aspect ratio α, and we can tell from the result as shown in Fig. 7 and Fig. 8 that the value of α will affect the final optimal result.

Then I set the α changes from 1.01 to 4 with 20 steps, the optimal result will be as follows:

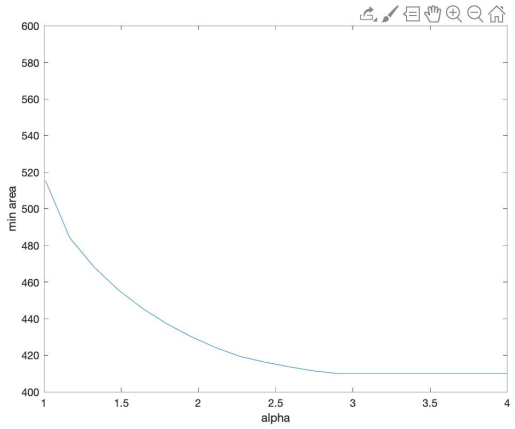

Fig. 9. Perturbation Analysis on the GP. α is the perturbed value and changes from 1.01 to 20.

From the Figure, we can easily tell that the minimum area obtained decreased when we loosen the aspect ratio constraint. At the upper left is the design corresponding to α_max = 1. In this case each of the blocks is fixed to be square. With a loose enough constraint on aspect ratio, the blocks can configure themselves so that they give a perfect packing; there is no unused space inside the bounding box. When the aspect ratio larger than 2.86, the optimal value will be just the sum of the blocks which is preferential.

VIII. Conclusion

In this Floor Planning Problem Project, we implemented two methods to solve. LP can obtain the result with square bounding box which more preferential in VLSI design and GP can obtain the result with more preferential aspect ratio of blocks. These two convex optimization approaches are both feasible in solving the Floor Planning Problem.

Acknowledgment

I would like to thank Dr. Kristin Dana for her feedback and support all throughout the course.

References

[1] Chaomin Luo, Miguel F. Anjos, Anthony Vannelli, 2008. Large-Scale Fixed-Outline Floorplanning Design Using Convex Optimization Techniques

[2] Boyd, S., Vandenberghe, L., 2004, Convex Optimization

[3] Boyd, S., Kim, S. J., Vandenberghe, L., Hassibi, A. 2007. A tutorial on geometric programming Optimization and engineering