Final Project - EEC254

Floor Planning via Convex Optimization

Ahmed Mahmoud

1 Abstract

2 Introduction

Floor planning problem tries to find the optimal position and/or dimensions of geometric objects (commonly rectangles) within a space such that there is no overlap between them. We refer to the objects as *cells* which are axis-aligned. There could be some constraints on the cells e.g., area-constraints, aspect ratio constraints, and positions constraints. The objective is usually to minimize the size (e.g., area, volume, perimeter) of the bounding rectangle or maximize the size of the cells. Motivation.

3 System Model

We consider N axis-aligned cells $(c_1, \cdots, c_N)$ in 2D each specified by its lower-left corner (x_i, y_i), height h_i, and width w_i. The cells are configured inside a bounding rectangle (BR) of height H, width W, and the origin as its lower-left corner. The cells are always desired to not overlap. This constraint makes the general floor-planning a complicated combinatorial optimization problem. However, if the relative positioning of the cells is specified, the floor-planning problem can be cast as convex optimization.

Variables

The variables of the optimization problem are the position and size of the cells (x_i, y_i, h_i, w_i for $i = 1, \cdots, N$) and the size of the bounding rectangle (H, W).

Objectives

There are two different objectives that the problem of floor-planning may try to achieve either of them

1. Minimizing the are of the bounding rectangle of fixed-size cells. Here we consider minimizing the perimeter of the bounding rectangle since the area is a non-convex function and its log is concave function.

2. Maximizing the area of the cells for a fixed-size bounding rectangle. We take log are of the cell i.e., $max(log(w_i)+log(h_i))$ for $i = 1, \cdots, N$. This is a multi-criterion problem with N objective function which can be scalarized by take the sum of all objective function, thus maximizing $\sum_{i=1}^{N}(log(w_i) + log(h_i))$.

Constraints

Some or all of the following constraints maybe considered in the floor-planning problem

- non-overlapping constraints: or **int**$(c_i \cap c_j) = \phi$, $i \neq j$. We achieve this by imposing relative position instead using two binary relations on the indices $\{1, \cdots, N\}$: $\mathcal{L}$ and $\mathcal{U}$ means *left of* and *under* respectively. Two relations are sufficient because the left/right and under/above are pairs of anti-symmetric, transitive relations. For example, if $(i, j) \in \mathcal{L}$, then c_i is left of c_j and thus $x_i + w_i \leq x_j$. To ensure full description of all relative positions, we require for each (i, j) with $i \neq j$, one of the following holds
$$(i,j) \in \mathcal{L} \qquad (j,i) \in \mathcal{L} \qquad (i,j) \in \mathcal{U} \qquad (j,i) \in \mathcal{U}$$
This will give $N(N-1)/2$ inequalities. Using the transitivity of the relation, the number of inequalities can be greatly reduced.

- minimum spacing constraints: To avoid tight packing, imposing a minimum psotive spacing ρ is possible. This changes the interpretation of $(i, j) \in \mathcal{L}$ to mean $x_i + w_i + \rho \leq x_j$.

- bounding constraints: $x_i \geq 0, y_i \geq 0, x_i + w_i \leq W, y_i + h_i \leq H$ for $i = 1, \cdots, N$.

- minimum cell area constraints: formulated as $log(w_i) + log(h_i) \geq log(A_i^{min})$ or $\sqrt{h_i w_i} \geq \sqrt{A_i^{min}}$ where A_i^{min} is cell i minimum area constraint.

- aspect-ratio constraints: aspect-ratio is the ratio of height over width. The constraint is $\omega_i^{min} \cdot w_i \leq h_i \leq \omega_i^{max} \cdot w_i$ where ω_i^{min} and ω_i^{max} is the minimum and maximum aspect ratio constraints for the cell i.

- alignment constraints: used for aligning two edges of two cells together. For example, the horizontal center of cell i aligns with the top of the cell j when $y_i + \frac{w_i}{2} = y_j + w_j$.

- symmetry constraints: cells maybe required to be symmetric about a vertical or horizontal axis that can be fixed or floating. For example, to specify that cells i and j to be symmetric about vertical axis x_{axis}, the constraint is $x_{axis} - (x_i + \frac{w_i}{2}) = x_j + \frac{w_j}{2} - x_{axis}$.

- similarity constraints: cell i maybe required to be α-scaled version of cell j for fixed α i.e., $w_i = \alpha w_j$, $h_i = \alpha h_j$.

- containment constraints: A cell maybe required to contain a certain point (x_p, y_p) inside the bounding rectangle such that $0 \leq x_p - x_i \leq w_i$ and $0 \leq y_p - y_i \leq h_i$. It maybe also required that the cell to be inside a given polyhedron.

4 Problem Formulation

Primal Problem

Here we formulate the primal and dual optimization problems of the floor planning. We will examine floor-planning problem in which we are trying to minimize the area of the bounding rectangle with no-overlapping, minimum area, and aspect ratio constraints.

The relative positions relations $\mathcal{L}$ and $\mathcal{U}$ can be placed by matrices L and U such that

$$L_{ij} = \begin{cases} 1 & (i,j) \in \mathcal{L} \\ 0 & \text{otherwise} \end{cases} \qquad U_{ij} = \begin{cases} 1 & (i,j) \in \mathcal{U} \\ 0 & \text{otherwise} \end{cases}$$

From that, we can write the primal convex problem in canonical form as:

$$\begin{array}{ll}\text{minimize} & 2(W+H) \\ \text{subject to} & -x \preceq 0, \quad -y \preceq 0, \quad -w \preceq 0, \quad -h \preceq 0, \quad -W \leq 0, \quad -H \leq 0 \\ & x + w - W \times \mathbf{1} \preceq 0, \quad y + h - H \times \mathbf{1} \preceq 0 \\ & A_i^{min}/h_i - w_i \leq 0 \quad i = 1, \cdots, N \\ & L_{ij} \times (x_i + w_i - x_j) \leq 0, \quad i, j = 1, \cdots, N \\ & U_{ij} \times (y_i + h_i - y_j) \leq 0, \quad i, j = 1, \cdots, N \\ & \textbf{diag}(\omega^{min} w^T) - h \preceq 0, \quad h - \textbf{diag}(\omega^{max} w^T) \preceq 0,\end{array} \tag{1}$$

where $\times$ represents scalar multiplication and $\cdot$ is dot-product.

The first set (line) of constraints ensures the solution is physically feasible (non-negative) while the second set ensures that everything lies inside the bounding rectangle. The third set is the minimum area constraints. The forth and fifth set are the relative position constraints. The final set is the aspect ratio constraints. Since L and U are sparse matrices, most of the inequalities from the forth and fifth set of constraints will be inactive.

We now introduce the Lagrangian multipliers in order to formulate the Lagrangian function. Let λ_i for $i = 1, \cdots, 13$ be the Lagrangian multipliers. $\lambda_i \in \mathbb{R}_+^N$ for $\{\forall i : i \neq 5, 6, 10, 11\}$, λ_5, $\lambda_6 \in \mathbb{R}_+$, and $\lambda_{10}, \lambda_{11} \in \mathbb{R}_+^{N \times N}$. Then, the Lagrangian function can be written as

$$\begin{aligned} L(W, H, x, y, w, h, \lambda_i) &= 2(W+H) - \lambda_1 \cdot x - \lambda_2 \cdot y - \lambda_3 \cdot w - \lambda_4 \cdot h - \lambda_5 \times W - \lambda_6 \times H \\ &+ \lambda_7 \cdot (x + w - W \times \mathbf{1}) + \lambda_8 \cdot (y + h - H \times \mathbf{1}) + \lambda_9 \cdot (A^{min}/h - w) \\ &+ \sum_{ij=1}^{N} \lambda_{10_{ij}} L_{ij} \times (x_i + w_i - x_j) + \sum_{ij=1}^{N} \lambda_{11_{ij}} U_{ij} \times (y_i + h_i - y_j) \\ &+ \sum_{i=1}^{N} \lambda_{12_i} \left(\omega_i^{min} \times w_i - h_i\right) + \sum_{i=1}^{N} \lambda_{13_i} \left(h_i - \omega_i^{max} \times w_i\right) \end{aligned}$$

After several rearrangement,

$$\begin{aligned} L(W, H, x, y, w, h, \lambda_i) &= W\left(2 - \lambda_5 - \mathbf{1}^T \cdot \lambda_7\right) + H\left(2 - \lambda_6 - \mathbf{1}^T \cdot \lambda_8\right) \\ &+ \sum_{i=1}^{N} x_i \left(\lambda_{7_i} - \lambda_{1_i} + \sum_{j=1}^{N} \lambda_{10_{ij}} \times L_{ij} - \sum_{j=1}^{N} \lambda_{6_{ji}} \times L_{ji}\right) + \sum_{i=1}^{N} y_i \left(\lambda_{8_i} - \lambda_{2_i} + \sum_{j=1}^{N} \lambda_{11_{ij}} \times U_{ij} - \sum_{j=1}^{N} \lambda_{7_{ji}} \times U_{ji}\right) \\ &+ \sum_{i=1}^{N} w_i \left(\lambda_{7_i} - \lambda_{3_i} - \lambda_{9_i} + \lambda_{12_i} \times \omega_i^{min} - \lambda_{13_i} \times \omega_i^{max} + \sum_{j=1}^{N} \lambda_{10_{ij}} \times L_{ij}\right) \\ &+ \sum_{i=1}^{N} \left(h_i \left(\lambda_{8_i} - \lambda_{4_i} - \lambda_{12_i} + \lambda_{13_i} + \sum_{j=1}^{N} \lambda_{11_{ij}} \times U_{ij}\right) + \frac{\lambda_{9_i} A_i^{min}}{h_i}\right) \end{aligned}$$

Dual Problem

To derive the dual objective function g, we minimize the Lagrangian function over the problem variables by taking the infimum over W, H, x, y, w, h, and λ_i. The terms multiplied by W, H, x, y and w are all linear. Thus, when taking the infimum, these terms should be zero otherwise the problem will be unbounded below. However, h's terms in non linear. To find it is infimum, we took the gradient and set it to zero and we get

$2\sum\sqrt{\lambda_{9_i}A_i^{min}\left(\lambda_{8_i}-\lambda_{4_i}+\sum\lambda_{ij}\times U_{ij}\right)}$. After several manipulation, we can re-write all terms in matrix format. Thus, the dual problem can be written in compactly as

$$
\begin{aligned}
\text{maximize} \quad & 2\sum_{i=1}^{N}\sqrt{\lambda_{9_i}A_i^{min}\left(\lambda_{8_i}-\lambda_{4_i}-\lambda_{12_i}+\lambda_{13_i}+\sum_{j=1}^{N}\lambda_{ij}\times U_{ij}\right)} \\
\text{subject to} \quad & 2-\lambda_5-\mathbf{1}^T\cdot\lambda_7=0 \\
& 2-\lambda_6-\mathbf{1}^T\cdot\lambda_8=0 \\
& \lambda_7-\lambda_1+\mathbf{diag}(\lambda_{10}L^T)-\mathbf{diag}(\lambda_6L^T)=0 \\
& \lambda_8-\lambda_2+\mathbf{diag}(\lambda_{11}L^T)-\mathbf{diag}(\lambda_7L^T)=0 \\
& \lambda_7-\lambda_3-\lambda_9+\mathbf{diag}(\omega^{min}\lambda_{12}^T)-\mathbf{diag}(\omega^{max}\lambda_{13}^T)+\mathbf{diag}(\lambda_{10}L^T)=0 \\
& \lambda_i\succeq 0
\end{aligned}
$$

Geometric Programming

In order to formulate the problem as geometric programming, few changes need to be done to 1. Geometric programming gives us greater flexibility in formulating the problem. We can minimize the area as WH since it is posynomial function. All constraints constraints can be easily recognized as posynomial functions. We integrate the relative positions constraints in the objective function. We show this with the example shown in Figure 1 that shows the relative position constraints [1]. Here, the height and width can be expressed as

$$W = max(w_A + w_B, w_C + w_D) \qquad H = max(h_A + h_B, h_C + h_D)$$

The optimization problem then becomes

$$\text{maximize} \quad WH = max(w_A + w_B, w_C + w_D)max(h_A + h_B, h_C + h_D)$$

The expression is a generalized posynomial with the variables $w_A, \cdots, w_D$ and $h_A, \cdots, h_D$ and from which 1 can be converted to geometric programming problem.

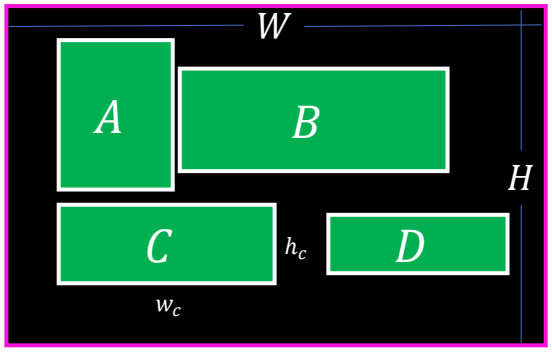

Figure 1: Illustrative configuration shows the relative position constraints for floor planning

Constraints

We take here a different turn where we eliminate the relative position constraints. This is common in practice especially in VLSI design were the area of the building cells are known, however, the relative position is not known or not important. The makes the problem much harder since now there is no easy way to express the non-overlap conditions. Impose arbitrary relative positions constraints may lead to sub-optimal solution

without good measure for how far the sub-optimal solution is from the optimal solution. Using brute-force won't scale well since the problem NP-hard [2].

To tackle this problem, we utilize a divide-and-conquer style algorithm to make the computation tractable. First, we divide the cells into two groups of two (almost-)equal areas based on their minimum area constraints. Second, one of the group is assigned to be left of all the cells of the other group and vice-versa. We do this recursively by diving each group into two sub-groups of (almost-)equal area and switch to assign the relative position as up-and-down. We keep iterating between direction of assignment until we reach an empty group. This can be seen as breadth-first search algorithm to construct L and U matrices.

5 Results

We implement the floor planning optimization problem without relative position constraints. Our implementation is done onto of CVX library in MATLAB. Dividing the cells into (almost-)equal groups is a *partition* problem for which we implemented a heuristics algorithm that operates by randomize the area array and then iterates over randomized array and assigns the processed area to the group with least sum. The algorithm takes $O(nlog(n))$ where n is the size of input area array. We compared our implementation using two models of 60 and 32 cells. The aspect ratio bounds are set to be between 0.5 and 2.0. We did our comparison against the solution given by Boyd et al. [2] as shown in Figure 2. Our solution have achieved better more tightly packed cells where the wasted area is 0% and 1% for the 32 and 60 cells respectively while Boyd et al. solution wasted area is 38% for both cases.

(a) Boyd - 60 Cells (b) Boyd - 32 Cells (c) Ours - 60 Cells (d) Ours - 32 Cells

Figure 2: Comparison between our relative-positions-constraints-free model and Boyd et al. [2] solution for 60 and 32 cells.

6 Conclusion

In this project, we showed the full formulation of the floor planning problem and how it can be solved as geometric programming problem while considering the majority of constraints taken into account in practice. Despite that we proposed to consider even more constraints, we found that considering less constraints is a problem that could happen more often in practice. We showed that it is better to have automated tool to assign the relative positions is better than assign them at random.

References

[1] Boyd, S., Kim, S.-J., Vandenberghe, L., and Hassibi, A. (2007). A tutorial on geometric programming. *Optimization and engineering*, 8(1):67.

[2] Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.