Chip Floor Planner

Muhammad Faisal

Outline

1. Problem Definition
2. Used Algorithm
3. Implementation aspect
4. Test Cases

Problem Definition

- Floor planning means the distribution of the different cells over the die area.

- Floor Planning Targets:
    - To minimize the used area and therefore the fabrication cost.

- Potential Floorplanning Constraints:
    - To put related cells near each other.
    - To put some cells nearer to the i/o Pads.

Problem Definition

- Specifically, the problem can be restated as the rearrangements of three types of rectangles in a given region:
    - The rectangles inside the core.
    - Rectangles of given area and flexible dimensions to be put in the core
    - The rectangles near the perimeter.

Used Algorithm

- The floor planning problem can be solved using a type of methods called linear programming.

- In this Method, the problem is written in the shape of matrix that when solved, a solution to the system is reached.

Used Algorithm

- The rectangles inside the core.
  - First, all the used variable to describe the dimensions and position should be greater than zero.
  - Each new rectangle increases the complexity of the problem as follows:
    - Two new variables are introduced to represent the position of the rectangle.
      - For these two variables, 2 new constraining relations are introduced.
    - Two new variables (for each previously existing rectangle) are introduced to represent the relative position between each two rectangles.
      - For each pair of two new variables, there are 4 constraining relations introduced.
        - These constraints are used to specify that there are no overlapping rectangles.

Used Algorithm

- The rectangles inside the core.
  - The equations and variables are as follows.

Minimize Y

Subject to

$$
\begin{array}{lll}
x_i \geq 0, & 1 \leq i \leq n & \\
y_i \geq 0, & 1 \leq i \leq n & \\
x_i + w_i \leq W & 1 \leq i \leq n & \\
y_i + h_i \leq Y & 1 \leq i \leq n & \\
x_i + w_i \leq x_k + W(x_{ik} + y_{ik}) & & 1 \leq i < j \leq n \\
y_i + h_i \leq y_k + H(1 + x_{ik} - y_{ik}) & & 1 \leq i < j \leq n \\
x_k + w_k \leq x_i + W(1 - x_{ik} + y_{ik}) & & 1 \leq i < j \leq n \\
y_k + h_k \leq y_i + H(2 - x_{ik} - y_{ik}) & & 1 \leq i < j \leq n
\end{array}
$$

Used Algorithm

- Rectangles of given area and flexible dimensions to be put in the core:
    - These are a more complex problem than putting a normal hard cell inside the core, since a new variable is now introduced that changes in order to choose the cell dimensions that best fit into the region with other hard cells.
    - However, since the relation between the width and length is not linear since their multiplication is a constant area.
        - The problem in this form is not a linear programming problem.
        - The problem can be turned again into a linear problem through linearization of the previous relation.
        - It is done through the following equations:

– First-order approximation

- $h_i = \Delta_i w_i + c_i$ $(y = mx+c)$
- $\Delta_i = (h_{i,min} - h_{i,max}) / (w_{i,max} - w_{i,min})$
- $c_i = h_{i,max} - \Delta_i w_{i,min}$

Used Algorithm

- Rectangles of given area and flexible dimensions to be put in the core:
  - After linearization the problem can be turned into:

Minimize Y
Subject to

$$
\begin{array}{ll}
x_i \geq 0, & 1 \leq i \leq n \\
y_i \geq 0, & 1 \leq i \leq n \\
x_i + w_i \leq W & 1 \leq i \leq n \\
y_i + (\Delta_i w_i + c_i) \leq Y & 1 \leq i \leq n \\
w_i \geq w_{i,min} & 1 \leq i \leq n \\
w_i \leq w_{i,max} & 1 \leq i \leq n \\
x_i + w_i \leq x_k + W(x_{ik} + y_{ik}) & 1 \leq i < j \leq n \\
y_i + (\Delta_i w_i + c_i) \leq y_k + H(1 + x_{ik} - y_{ik}) & 1 \leq i < j \leq n \\
x_k + w_k \leq x_i + W(1 - x_{ik} + y_{ik}) & 1 \leq i < j \leq n \\
y_k + (\Delta_k w_k + c_k) \leq y_i + H(2 - x_{ik} - y_{ik}) & 1 \leq i < j \leq n
\end{array}
$$

Used Algorithm

- The rectangles near the perimeter.
  - After doing the previous placing, it can be determined whether or not the die is core constrained:
    - Core constrained; perimeter rectangles can all be put around the core without resizing
    - I/O pads Constrained, the core width/length must be modified to allow for placing these rectangles.

Implementation aspect

- The project is built in C++ in a linux environment.
- The project uses a C++ library, called mipcl, to have an interface for a linear programming solver.
  - http://www.mipcl-cpp.appspot.com/documentation.html
- The Readme file contains all the input file constraints.
- The Readme file has getting started section to replicate the results of the test cases.
- The source code, test cases, and their output can be found in my github repo
  - https://github.com/mickey-me/floor-planning

Test Cases

- There are 13 test cases that were used to test the program that contains:
  - Around 3 test cases files for each type of rectangles alone.
  - 4 test cases files that have a combination of the different types of the inputs