Open Access Library Journal
2020, Volume 7, e6218
ISSN Online: 2333-9721
ISSN Print: 2333-9705

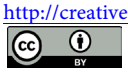
Standard Cell Placement Optimization Using Quadratic Placement Algorithm

Suren Abazyan[1], Narek Mamikonyan[1], Vakhtang Janpoladov[2]

[1]Yerevan State University, Yerevan, Armenia
[2]Russian-Armenian University, Yerevan, Armenia
Email: su.abazyan@gmail.com

How to cite this paper: Abazyan, S., Mamikonyan, N. and Janpoladov, V. (2020) Standard Cell Placement Optimization Using Quadratic Placement Algorithm. *Open Access Library Journal*, **7**: e6218.
https://doi.org/10.4236/oalib.1106218

Received: March 11, 2020
Accepted: April 5, 2020
Published: April 8, 2020



Abstract

Designs including tens of millions of standard cells in one chip are commonly used in current IC projects, so finding optimal location on a chip surface for each logic cell is a very important step in IC design. Apart from finding room for logic cell placement with minimum chip area, length of connecting wires is also playing big role and needs to be taken under control. In this paper, research and implementation of standard cell placement-optimizations' quadratic algorithm is described. Main research is on runtime and wire length. For 5K standard cells, algorithm implementation takes 83 second.

Subject Areas

Computer Engineering, Electric Engineering, High Performance Computing, Nanometer Materials, Software Engineering

Keywords

Placement, Optimization, Physical Design, Quadratic Placement

1. Introduction

Placement is the process of determining positions of standard cells and macros which has been synthesized. This is one of the most important and critical steps in IC design. Following few reasons are showing why optimal placement of standard cells is dramatically important for having good IC:

- Performance of a circuit—As technologies are becoming smaller, interconnect delays are increasing. This is leading to big power consumption. Hence, good placement of standard cells has huge impact on design performance
- Power disproportion—Badly placed cells can cause non-equal power consumption leading to congested heat point formulation, which can be source

of many problems even damaging full chip functionality.

- Routability—Placement can cause routibility issues later on routing stage and be reason of long wires.

To achieve optimal placement of standard cells with standard placement algorithm, in every iterate cells, which need to be moved, will be moved to the most optimal place in that iteration. Ideally, this process must be repeated until all standard cells are placed in their proper places.

There exist multiple ways of reducing placement runtime for standard cell placement [1] [2] [3]. Most of the existing algorithms have complex implementation [4].

As placement process is dealing with millions of macros and standard cells, it is very complex computational process. If considering that each cell can have multiple connections with other cells, computation complexity is increasing more and more.

The above-mentioned process will bring to high runtime with big designs (standard cell count > 5000). The most time-consuming part is the starting period where all connectivity for each standard cell mostly placed in (0, 0) coordinate. Above described algorithm will overcome the connectivity-based time consumption by logical excluding the cells connectivity which forces to cell stay near (0, 0) coordinates. The end part of the placement placer can make wary small movement which will not make huge impact in the standard cell placement.

To overcome upper mentioned difficulty, placement process is being divided into a few manageable steps—global placement, legalization, detailed placement.

This paper described algorithm can be used in global placement step. This is a very important step of placement as it has most impact on performance of IC.

The other two steps are for making placement of cells legal on placement grid and meeting placement constrains [5].

In the global placement, small movement of standard cell will be ignored for speeding up placement process.

2. Global Placement

Global placement is the step of finding optimal placement of circuit modules, based on different factors like wire length, area, parasitic etc.

It is a challenge to design efficient placement algorithms for producing high quality placement solutions of circuits with millions of cells [6].

Standard cell placement overall procedure using Hierarchical Alternating Linear Ordering can be described with **Figure 1**.

Algorithm is taking cell connectivity as its input and calculates mass center of cells. After some iterations, in-row placement is being performed for cells.

Standard cell connectivity netlist to in-row placement ordering can be simply described in **Figures 2(a)-(d)**.

In **Figure 2(a)**, initial netlist is given. This is the gate level netlist for placer. After gate-level netlist is read, placer places gates linary—keeping minimum wire length. After that is done, two virtual (left and right) groups are created, in

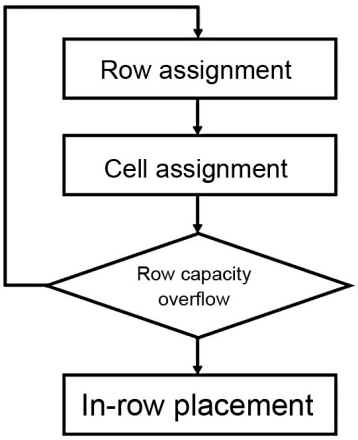


Figure 1. Hierarchical alternating linear ordering procedure flow.

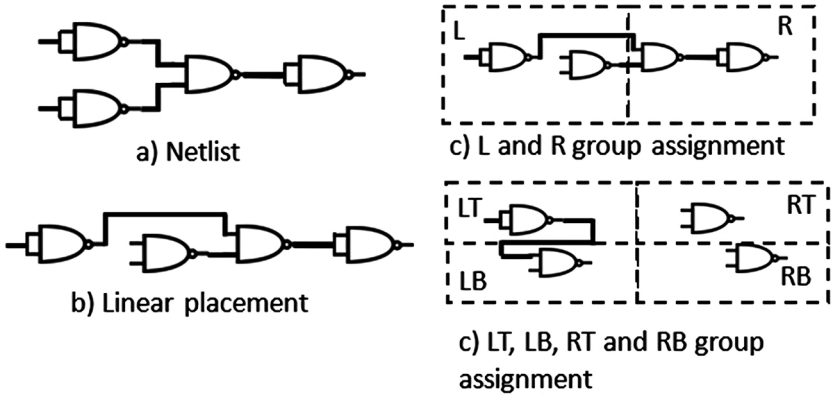


Figure 2. Standard cell connectivity netlist to in-row placement ordering.

which cells have approximately same sum of area (**Figure 2(c)**). After left and right groups are created, these are also being divided into small sub-groups (**Figure 2(c)**). This vary sum of area is also being kept equal for sub groups.

These iterations are going till all cells are assigned and constraint-based assigned to rows.

3. Quadratic Placer Algorithm Implementation

The input for the placer is netlist with initial fixed cells and their positions and the connections between placeable cells. Fixed and placeable cells constitute the vertices of a graph.

One more input for the algorithm is the move threshold, which described optimization level and is very important for algorithm runtime. This value is the specification of movement—the maximum value that displacement can have. In some cases, when threshold value is too small, placer can have multiple minor movements leading to quasi-infinite loops, which will bring to huge runtime.

Basic syntax of input netlist is shown in **Figure 3**.

After netlist is read, netlist handler is translating syntax to the graph-based connection.

```
Threshold = 0.2
FixCell_1 = {x1,y1}
FixCell_2 = { x2,y2}
....
FixCell_n = { xn,yn}
MoveCell_1 = { X1,Y1}
....
MoveCell_n = { Xn,Yn}
....

Connect_1 = {FixCell_1/PinA, FixCell_2/PinB}
Connect_2 = {FixCell_2/PinC, FixCell_4/PinA}
......
Connect_n = {FixCell_N/PinX, FixCell_M/PinY}
....
```

Figure 3. Input netlist syntax example.

At the beginning fixed cells are placed at their location and all movable cells are located in the start point of calculation system. Basic view of placement after netlist is read, can be shown in **Figure 4**.

Figure 4 is missing connections between all cells just to be clean. They will be shown later.

After Netlist is read and initial placement is done for fixed cells, algorithm is looping through all cells moving one cell at a time. First iterations are creating very rough estimation of cell placement, because in placement there are cells on (0, 0) position, and they are excluded from calculations.

To ensure near-optimal placement at the beginning stage of placement, algorithm is not considering cells which are placed at (0, 0) position. This approach is also minimizing runtime with a smaller number of needed calculations.

For algorithm run speed increasing and adding level of optimization algorithm is assigning initial values for cell group mass center. Cell groups are selected based on connections' count. After initial placement is done, mass center coordinates are updated.

For each cell group algorithm is first placing cells in row and starting optimization by dividing placement area in that way, that count of connections is remaining similar for each group. This way algorithm is keeping area of cell and routes similar for each group (approximated that all cells have same size).

Visualization of first few steps is shown in **Figures 5(a)-(d)**.

Last iteration is when all placeable cells are placed (**Figure 6** Step N). Main point is that every time when any cell is being placed, it is becoming "quasi-fixed" for next iteration's cell, so next cell which relates to that cell will have one more fixed connection and more accurate mass center. In other words, Placer considers only sells which have connection with fixed or quasi-fixed cells and ignores the connectivity of the standard cell which is leaded from (0, 0) coordinate.

Later, next iterations are assuming all other cells are placed, and moving one cell to meet minimum wire length.

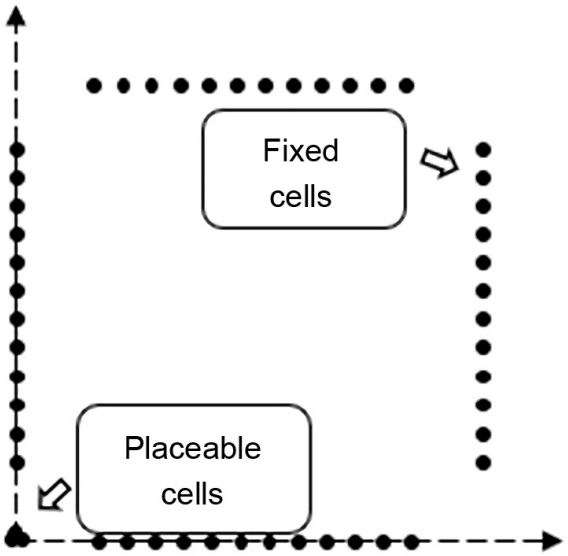


Figure 4. Placement after first iteration.

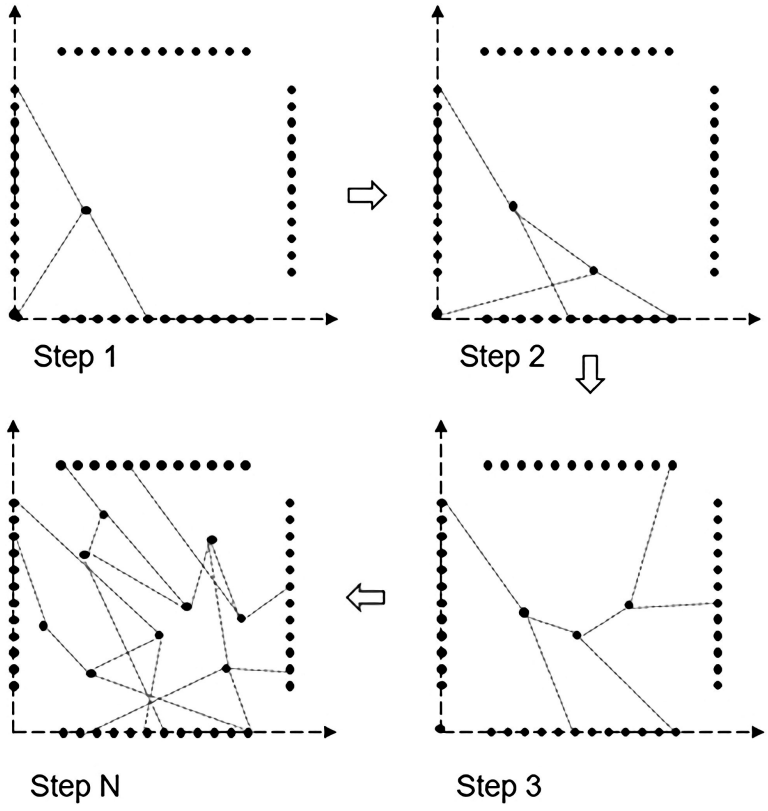


Figure 5. Visualization of first few steps of placement.

Placement is done, when for one iteration all cells are not being moved more than the threshold value. The threshold value implementation brings reduction of run time in the last finalizing part of the global placement. In this time, we have well placed and optimized placement, with optimal wire length estimation.

Algorithm block diagram is shown in **Figure 6**.

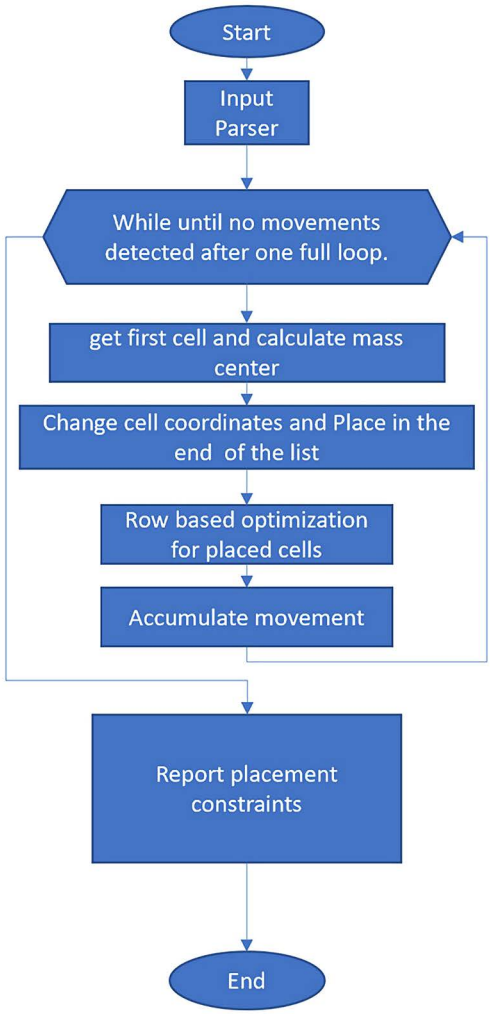


Figure 6. Algorithm block diagram.

4. Conclusions

In this paper, standard cell global placement algorithm is demonstrated. The approach is simple in implementation. Imbanded standard cell connectivity quasi-decreasing method brings algorithm implementation runtime degree to the level of existing algorithms ([1] [2] and [3]). For example, for logic which contains 5K standard cells, algorithm implementation takes 83 second.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Chan, T., Cong, J., Kong, T. and Shinnerl, J. (2000) Multilevel Optimization for Large-Scale Circuit Placement. *IEEE/ACM International Conference on Computer-Aided Design*, San Jose, CA, 5-9 November 2000, 171-176.

[2] Chan, T., Cong, J., Kong, T., Shinnerl, J. and Sze, K. (2003) An Enhanced Multilevel

Algorithm for Circuit Placement. *IEEE/ACM International Conference on Computer-Aided Design*, San Jose, CA, 9-13 November 2003, 299-305. https://doi.org/10.1109/ICCAD.2003.159704

[3] Huang, D.J.-H. and Kahng, A.B. (1997) Partitioning Based Standard Cell Global Placement with an Exact Objective. *ACM/IEEE International Symposium on Physical Design*, Napa Valley, CA, April 1997, 18-25. https://doi.org/10.1145/267665.267674

[4] Agnihotri, A.R. and Madden, P.H. (2007) Fast Analytic Placement Using Minimum Cost Flow. *Asia and South Pacific Design Automation Conference*, Yokohama, 23-26 January 2007, 128-134. https://doi.org/10.1109/ASPDAC.2007.357974

[5] Chen, J. and Zhu, W. (2012) An Analytical Placer for VLSI Standard Cell Placement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **31**, 1208-1221. https://doi.org/10.1109/TCAD.2012.2190289

[6] Chu, C. (2008) "Chapter 11: Placement" in Electronic Design Automation: Synthesis Verification and Testing. Elsevier/Morgan Kaufmann, San Francisco, CA.