A Standard-Cell Placement Tool for Designs with High Row Utilization [*]

Xiaojian Yang Bo-Kyung Choi Majid Sarrafzadeh

Computer Science Department, University of California, Los Angeles, CA 90095

xjyang,bkchoi,majid@cs.ucla.edu

Abstract

In this paper we study the correlation between wirelength and routability for standard-cell placement problem, under the modern place-and-route environment. We present a placement tool named Dragon *(version 2.1), and show its ability to produce good quality placement for designs with high row utilization. Compared to an industrial placer and an academic state-of-the-art placer, Dragon can produce placement with better routability and shorter total wirelength. We describe many novel algorithmic details and implementation details of this placement tool. Experimental results show that minimizing wirelength improves routability and layout quality.*

1. Introduction

Standard-cell placement problem has drawn extensive research attention in VLSI CAD area. One common classification for traditional placement methods is to put them into four basic categories: min-cut placement [1, 2], simulated annealing [3], analytical method [4, 5], and force-directed approach [6]. However, recently proposed placement tools rarely reside in any one of these categories. Most of them are more or less hybrid models[1]. Four classical techniques, plus clustering and flow-based method, frequently appear in these relatively new placement algorithms [10, 11, 12, 13, 14, 15, 16, 17, 18]. In addition to the above approaches that address half-perimeter wirelength, many techniques are proposed for timing [19, 20, 12, 21, 22, 23] and congestion [24, 25, 26] optimization. Most of them are based on wirelength minimization.

Wirelength is the fundamental objective in standard-cell placement problem. It is generally believed that a timing or congestion oriented approach can hardly be successful without a good wirelength minimization engine. The idea of timing driven placement is to reduce the wirelengths on certain paths instead of the total wirelength. A placement with shorter total wirelength is relatively easier to be modified to meet timing constraints. Similarly, a good placement with optimized wirelength has a higher probability that its congested regions are relatively smaller or less serious.

Our work is based on previous work of [14], in which a placement frame work and the corresponding implementation are described. The new contributions in this paper are: (a) We propose new algorithmic details that improve the placement quality, and discuss details of the implementation. (b) Unlike the previous work that focuses on estimated wirelength in placement, in this work, we study the relationship between wirelength and routability in a real place-and-route enviroment. The placement quality is evaluated by the layout quality after global and detailed routing.

The remainder of this paper is organized as follows. We introduce the framework of our standard-cell placement tool in Section 2. In Section 3, we explain the relevant aspects of implementation, and present novel algorithmic details. Section 4 shows the good quality of the placement tool by comparing it with an industrial tool and an academic tool. We conclude in Section 5.

2. Framework of Our Placement Tool

The main placement flow consists of two parts: recursive bisection and simulated annealing. These two techniques appeared very early in the literature of standard-cell placement and have proven effective. The recent advances in multilevel partitioning [27, 28] and their implementations imposed effects on the placement research in academia. Several placement tools [7, 14, 8] were proposed based on them.

The given circuit is recursively partitioned along alternatively horizontal and vertical cut line. The subcircuits after partitioning are assigned to rectangular bins. At some points a bin-based simulated annealing (i.e. the moving objects are the subcircuits in the bins) is performed to improve the current placement. Such a procedure terminates when certain stop criteria (e.g. average number of cells per bin is less than a given number) are satisfied. An adjustment step is then executed to fit the current bin-based placement into row structures. The next step is a cell-based simulated annealing. The bin structure still exists and the cells are moved between the centers of bins. The locations of these centers can be changed during the annealing procedure. After that, the final step simply spreads overlapped cells, and makes local improvement to obtain the detailed placement.

3. Detailed Implementations

In this section, we present details of implementation for this placement flow. Due to the space limitation, not every detail is showed with support of experiment data. However, all of them are real problems we have encountered and our conclusion comes from many experimental runs.

3.1 Terminal Propagation

Terminal propagation is the essential technique to the success of min-cut type placement. It leads to better bisection results for placement compared to the *pure* partitioning. This is because terminal propagation uses geometrical information of external terminals.

However, in our placement flow that combines partitioning and simulated annealing, terminal propagation is not as important as it is in min-cut placement flow. The reason is the following. In pure partitioning, although wrong decisions for cells are likely to be made without consideration of external terminals, these mistakes can be fixed in the later annealing stages. In addition, ignoring external connections may entail partitioner to focus on internal connections, leading to a better partitioning solution.

We conduct the following experiments to study terminal propagation in both our flow and min-cut placement flow. Table 1 lists the final wirelength comparison for different terminal propagation usages. Experiments show that in our placement flow, it is better not to start terminal propagation too early or too late. Some medium levels are more reasonable points to start using terminal propagation.

3.2 From Bins to Rows

An inevitable problem for the bin based approach is the difference between the number of rows in the design and the number of rows in the bin grids[2]. Because of this limitation, previous experiments of bin based approach were performed by the aid of detailed placer (e.g., [15]),

[*]This work was supported by NSF under Grant #CCR-0090203

[1]Some exceptions are [7, 8] (pure min-cut) and [9] (analytical, with flow method in detailed placement).

[2]Unbalanced partitioning (used in min-cut placement) does not apply here, because bin annealing requires that all bins have roughly the same size.

test circuit 1		test circuit 2	
level	*wirelength*	*level*	*wirelength*
1	4.316	1	4.560
2	4.328	2	4.521
3	4.322	3	4.608
4	4.264	4	4.508
5	4.305	5	4.503
6	4.311	6	4.582
7	4.329	7	4.492

Table 1: Wirelength and runtime comparison for placement using different terminal propagation strategies in partitioning. *level* **indicates at which placement level we start using terminal propagation in partitioning. Wirelengths are in meters. Every entry is the average value from three placement runs.**

or on the benchmarks that have 64 or 128 rows (e.g., [14]). We devise a simple-but-effective adjustment step and put it before the cell annealing process. The basic idea is to merge all the cells within the same column in the bin grids, and then evenly divide them into rows.

This adjustment step does not consider connections between cells, rendering quality loss of wirelength. In experiments we observed about 3% worse wirelength after this adjustment step A flow-based algorithm could be used for this specific problem and lead to better solution. However, minor loss of quality in this step is acceptable since the following cell annealing will cover the loss.

3.3 Cell Annealing with Bin Structure

We propose a new approach at the cell annealing stage. Specifically, we allow cell moves between bins while changing the centers of bins. Fig. 1 explains the difference between this method and previous approaches. In the figure, (a) and (b) were used in [19]; (c) was used in [11, 14]; (d) is the new method in our approach.

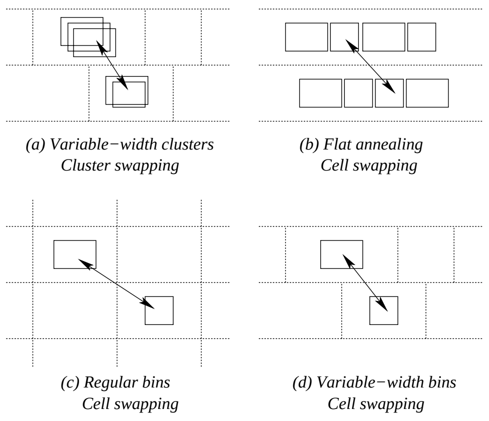

Fig. 1: Comparison between different move types in simulated annealing. (a),(b),(c) are previously used methods and (d) is our new approach.

In (a), cell clusters, not cells are swapped during simulated annealing. The freedom of cells are confined by clusters. In (b), cells are next to each other. Moving a cell will change all the locations of cells on the right. Although the authors in [19] employed wirelength estimation technique, this flat annealing is still the most time consuming part. (c) was used in [11, 14]. Its drawback is that all the bins have the same width while every bin has different total cell width. This problem becomes serious especially when the average number of cells per bin is small, or the cell widths vary considerably. In this case the improved wirelength does not correlate to the true wirelength after spreading cells. (d) is new method in our approach. The idea is to keep cells overlapped at bin centers for speeding up cost evaluation process. However, the bin widths are not fixed. The center of a bin will be updated periodically in the simulated annealing[3], according to the summation of bin widths for all the bins on the left. As the temperature becomes lower, less moves are accepted, thus the changes of bin widths become smaller. This procedure of minimizing wirelength will converge at the end. The overlapped placement obtained by this method correlates well to the placement after spreading out cells, providing a good initial point for detailed placement.

It should be noted that the method of variable-width bins could be extended to low-density standard-cell designs. White space, or feedthrough area can be assigned into bins and the simulated annealing approach can still be applied. The only difference is that the bins are wider now — it contains not only the cells but also the white space. Detailed placement process needs to be modified accordingly.

3.4 Balance Control

Control of the maximum row length is a very important topic for designs with high row utilization. A gradual budget assignment approach was proposed in [16] on this problem. In our placement flow, the row unbalance comes from the inexact bisections and bin annealing. It is well-known that low tolerances of partitioning result in suboptimal objectives. Moreover, due to the accumulation of the unbalance for a series of partitionings, it is extremely hard to control the row balance by lowering the tolerance in partitioning. Similarly, in the bin annealing stage, banning the cluster moves that violate row balance substantially confines the freedom of clusters and results in loss of placement quality.

The bin adjustment method in Section 3.2 partially helps reducing the row unbalance, yet it cannot eliminate the unbalance. The author in [9] uses a network flow based algorithm to solve the balancing problem. We simplified the flow model in which cells can only be moved between adjacent rows and implemented the similar approach.

The cell annealing stage provides a good opportunity to control the maximum row length. There are two ways to achieve this objective: penalizing the overflowed rows, or disallowing moves that violates row balance. The former needs fine tuning of simulated annealing for appropriate coefficients, while the latter is relatively easy. According to our simplicity principle, we adopt the second approach in our work.

Another detail for the implementation of cell annealing is the choice of move types. We allow both cell swapping and cell shifting, i.e., moving a cell from one bin to another. Experiments show that introducing cell shifting not only improves wirelength results, but also greatly helps the balance control. Moreover, our experience indicates that the flow-based row adjustment method is unnecessary in our flow — cell based annealing solves the balance problem well.

In our experiments, we observed that balance control for very tight design (e.g., 0.01% white space) is very difficult and usually leads to significant loss of quality. Considering that this very tight design is less relevant with real designs, we do not further discuss balance control problem for very tight designs.

3.5 Spreading Cells

At the beginning of the final placement stage, we face the problem of spreading the cells within the bins. The authors in [30] use the optimal placer for small placement instances. We integrated the same branch-and-bound algorithm to spread cells in bins with less than 8 cells. However, we found that this step is unnecessary in our placement flow, as the later local improvement covers the difference between an optimal spread and a random spread.

Table 2 shows that the gain from optimal spreading cells is shadowed by the later local improvement step. This is not the first time we have met the situation: an optimization at a given step may not be necessary due to the following optimizations. We hope that the experience obtained from experiments will be helpful for understanding placement problem in a global view.

[3]For example, before the temperature change.

stage	test circuit 1		test circuit 2	
	random spreading	optimal spreading	random spreading	optimal spreading
after cell annealing	4.50		4.06	
after spreading	4.68	4.55	5.20	4.79
after local impr.	4.52	4.52	4.46	4.41
Impr. at last step	3.4%	0.7%	14.2%	7.9%

Table 2: Comparison of final placement wirelengths using random spreading or optimal spreading. Although optimal spreading gives better wirelength at this step, the final wirelength after local improvement step is similar to that of random spreading.

4. Experimental Results

In order to create a set of benchmarks with correct routing information, we scale circuits in IBM-PLACE to match the standard-cell sizes in a 0.18μm library, which was obtained from Artisan Components Inc. through the academic research support program. We then output a pair of LEF/DEF files and use an industrial floorplanner to decide the core size and rows. Another source of benchmarks are from ISPD01 benchmarks suits[22]. These benchmarks are in structural verilog file format. We use an industrial synthesis tool to compile them with 0.18μm standard-cell library to create LEF/DEF files. The circuits size range from 3,000 cells to 66,700 cells. The row utilization of all the circuits are more than 98%. Four or six routing layers are used for the benchmarks.

We compared our placement tool with a well-known industrial placer, Cadence QPlace (Silicon Ensemble, Version 5.3), and a state-of-the-art academic placer, Capo (September 2001 version).[4] All three placers read the same LEF/DEF files and output placement results in DEF format. We then use Cadence WarpRouter to read the placement outputs and do global and final routing. We consider the routing result *success* (no violation), *finished* (with a small number of violations) or *failure* (too many violations or out of time). Experimental result summary[5] in Table 3 shows that Dragon produces better placement in terms of routability. Also Dragon produces layout with shorter wirelength and smaller number of vias (measured after global and detailed routing by Cadence WarpRoute).

Placer	Successful routing	Finished routing	Failed routing
QPlace	8	4	0
Capo	7	2	3
Dragon	10	2	0

Table 3: Comparison between Cadence QPlace, Capo and Dragon on 12 circuits. We consider the routing result as *success* (no violation), *finished* (with a small number of violations) or *failed* (too many violations or out of time).

5. Conclusion

The main idea of this paper is a simple-but-good placer for wirelength minimization. We have shown that minimizing wirelength is still the important topic for routability, even in modern fixed-die context. We hope that the simplicity of the placer can help future studies on more complex issues in placement process, such as meeting timing constraints.

6. Acknowledgments

The authors would like to thank the reviewers for their helpful comments. Special thanks to Prof. Igor Markov whose idea of comparing routability for different placers under the same context leading to this work.

[4]We do not compare the work in [14] (Dragon2000) because: (a) Dragon2000 can not read LEF/DEF files, and (b) Dragon2000 has the limitation on the number of rows.

[5]Due to the length limitation, we only report a summary of the experimental results in this section. Please refer to Dragon website(http://er.cs.ucla.edu/Dragon) for full experimental results.

7. References

[1] M. A. Breuer. "A Class of Min-cut Placement Algorithms". In *Design Automation Conference*, pages 284–290. IEEE/ACM, 1977.

[2] A. E. Dunlop and B. W. Kernighan. "A Procedure for Placement of Standard Cell VLSI Circuits". *IEEE Transactions on Computer Aided Design*, 4(1):92–98, January 1985.

[3] C. Sechen and A. Sangiovanni-Vincentelli. "TimberWolf3.2: A New Standard Cell Placement and Global Routing Package". In *Design Automation Conference*, pages 432–439. IEEE/ACM, 1986.

[4] J. M. Kleinhans, G. Sigl, F. M. Johannes, and K. J. Antreich. "GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization". *IEEE Transactions on Computer Aided Design*, 10(3):365–365, 1991.

[5] G. Sigl, K. Doll, and F. M. Johannes. "Analytical Placement: A Linear or a Quadratic Objective Function". In *Design Automation Conference*, pages 427–432. IEEE/ACM, 1991.

[6] S. Goto and E. S. Kuh. "An Approach to the Two-Dimensional Placement Problem in Circuit Layout". *IEEE Transactions on Circuits and Systems*, 25(3):208–214, 1978.

[7] A. E. Caldwell, A. B. Kahng, and I. L. Markov. "Can Recursive Bisection Alone Produce Routable Placements?". In *Design Automation Conference*, pages 477–482. IEEE/ACM, June 2000.

[8] M. C. Yildiz and P. H. Madden. "Improved Cut Sequences for Partitioning Based Placement". In *Design Automation Conference*, pages 776–779. IEEE/ACM, 2001.

[9] Jens Vygen. "Algorithms for Large-Scale Flat Placement". In *Design Automation Conference*, pages 746–751. IEEE/ACM, 1997.

[10] D. Huang and A. B. Kahng. "Partitioning-based Standard-cell Global Placement with an Exact Objective". In *International Symposium on Physical Design*, pages 18–25. ACM, April 1997.

[11] M. Sarrafzadeh and M. Wang. "NRG: Global and Detailed Placement". In *International Conference on Computer-Aided Design*. IEEE, November 1997.

[12] H. Eisenmann and F. M. Johannes. "Generic Global Placement and Floorplanning". In *Design Automation Conference*, pages 269–274. IEEE/ACM, 1998.

[13] X. Yang, M. Wang, K. Eguro, and M. Sarrafzadeh. "A Snap-On Placement Tool". In *International Symposium on Physical Design*, pages 153–158. ACM, April 2000.

[14] M. Wang, X. Yang, and M. Sarrafzadeh. "Dragon2000: Fast Standard-cell Placement for Large Circuits". In *International Conference on Computer-Aided Design*, pages 260–263. IEEE, 2000.

[15] T. F. Chan, J. Cong, T. Kong, and J. R. Shinnerl. "Multilevel Optimization for Large-Scale Circuit Placement". In *International Conference on Computer-Aided Design*, pages 171–176. IEEE, 2000.

[16] K. Zhong and S. Dutt. "Effective Partition-Driven Placement with Simultaneous Level Processing and Global Net Views". In *International Conference on Computer-Aided Design*, pages 171–176. IEEE, 2000.

[17] S. Hur and J. Lillis. "Mongrel: Hybrid Techniques for Standard Cell Placement". In *International Conference on Computer-Aided Design*, pages 165–170. IEEE, 2000.

[18] O. Faroe, D. Pisinger, and M. Zachariasen. "Local Search for Final Placement in VLSI Design". In *International Conference on Computer-Aided Design*, pages 565–572. IEEE, 2001.

[19] W. Swartz and C. Sechen. "Timing Driven Placement for Large Standard Cell Circuits". In *Design Automation Conference*, pages 211–215. IEEE/ACM, 1995.

[20] M. Sarrafzadeh, D. A. Knol, and G. E. Tellez. "Unification of Budgeting and Placement". In *Design Automation Conference*, pages 758–761. IEEE/ACM, 1997.

[21] S. L. Ou and M. Pedram. "Timing-driven Placement Based on Partitioning with Dynamic Cut-net Control". In *Design Automation Conference*, pages 472–476. IEEE/ACM, June 2000.

[22] Y. C. Chou and Y. L. Lin. "A Performance-Driven Standard-Cell Placer Based on a Modified Force-Directed Algorithm". In *International Symposium on Physical Design*, pages 24–29. ACM, April 2001.

[23] B. Halpin, C. Y. Chen, and N. Sehgal. "Timing Driven Placement using Physical Net Constraints". In *Design Automation Conference*, pages 780–783. IEEE/ACM, 2001.

[24] P. N. Parakh, R. B. Brown, and K. A. Sakallah. "Congestion Driven Quadratic Placement". In *Design Automation Conference*, pages 275–278. IEEE/ACM, June 1998.

[25] M. Wang, X. Yang, and M. Sarrafzadeh. "Congestion Minimization During Placement". *IEEE Transactions on Computer Aided Design*, 19(10):1140–1148, 2000.

[26] X. Yang, R. Kastner, and M. Sarrafzadeh. "Congestion Reduction During Placement Based on Integer Programming". In *International Conference on Computer-Aided Design*, pages 573–576. IEEE, 2001.

[27] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. "Multilevel Hypergraph Partitioning: Application in VLSI Domain". In *Design Automation Conference*, pages 526–529. IEEE/ACM, 1997.

[28] C. J. Alpert, J. H. Huang, and A. B. Kahng. "Multilevel Circuit Partitioning". In *Design Automation Conference*, pages 530–533. IEEE/ACM, 1997.

[29] W. J. Sun and C. Sechen. "Efficient and Effective Placement for Very Large Circuits". *IEEE Transactions on Computer Aided Design*, 14(3):349–359, March 1995.

[30] A. E. Caldwell, A. B. Kahng, and I. L. Markov. "Optimal Partitioners and End-case Placers for Standard-cell Layout". *IEEE Transactions on Computer Aided Design*, 19(no.11):1304–1314, Nov 2000.