Satisfying Whitespace Requirements in Top-down Placement

Jarrod A. Roy, David A. Papa, Aaron N. Ng, Igor L. Markov
University of Michigan, EECS Department, Ann Arbor, MI 48109-2121
{royj,iamyou,aaronnn,imarkov}@umich.edu

ABSTRACT

In this invited note we outline several algorithms and features appearing in Capo 10, free open-source software for congestion-driven standard cell placement, mixed-size placement and floorplanning. Capo scales on par with industry placers and has been successfully used with a broad range of netlists. It can also satisfy lower bounds on local whitespace, using several techniques for global, detail and macro placement.

Categories and Subject Descriptors

J.6 [**Computer-Aided Engineering**]: Computer-aided design (CAD)

General Terms

Algorithms, Design

Keywords

Physical Design, Placement, Floorplanning

1. INTRODUCTION

Management of whitespace is a key issue in physical design as it has a profound effect on the quality of a placement. The literature includes several techniques to optimize whitespace distributions. Achieving uniform whitespace allocation in top-down placement was introduced and analyzed in [5]. In [2], analytical placement is used to allocate whitespace in sparse designs. Non-uniform whitespace allocation by adding free cells to a design was presented in [1]. There are fewer techniques for respecting whitespace constraints imposed by a designer while still optimizing interconnect. Such constraints can be helpful as respecting them can improve routability, allow for effective buffer insertion, etc.

One trivial way to ensure sparser cell densities in a placement is by artificially increasing cell sizes before placement (*bloating*) and shrinking them back to normal size afterward. For the bloating to be effective, the majority of the original whitespace of the design must be taken up by the bloating. This reduces the amount of whitespace available to the placer, which complicates the work of many placement algorithms, increasing interconnect length or leading to overlapping circuit modules [7].

This note outlines the methods used in the open-source academic placer Capo 10 for satisfying whitespace constraints in top-down placement. Section 2 describes Capo's whitespace allocation techniques and illustrates how they can be used to satisfy given placement density constraints. Whitespace allocation in detail placement and mixed-size placement are discussed in Sections 3 and 5 respectively. Capo's whitespace techniques for improving routability are examined in Section 4 and the effect of whitespace management on placement memory usage is addressed in Section 6.

ISPD'06, April 9–12, 2006, San Jose, California, USA.
ACM 1-59593-299-2/06/0004.

2. WHITESPACE ALLOCATION

Capo 10 is a min-cut placer whose main algorithms are described in [9, 10, 11]. Capo placement proceeds by successively dividing *placement bins*, the first of which contains the entire core area and all movable objects, until the bins are small enough to be optimally placed. Whitespace allocation in Capo is done per placement bin: either *uniform*, *minimum local* or *safe* whitespace allocation is chosen based on the bin's whitespace and user-configurable options.

Uniform Whitespace. If a bin has a user-defined "small" amount of whitespace or less, partitioning attempts to divide the cell area approximately in half, within a given tolerance. The appropriate partitioning tolerance, derived in [5], is based on whitespace degradation: the phenomenon that discreteness in partitioning and placement does not allow for exact uniform whitespace distribution. After a partitionment (i.e., a partitioning solution) is computed, the geometric cutline for the bin is positioned so that each side of the cutline has an equal percentage of whitespace. As tolerance is calculated assuming a fixed cutline, the cutline is shifted to make whitespace more uniform. Such whitespace allocation generally produces routable placements, at the cost of increased wirelength.

Minimum Local Whitespace. If a bin has more than a user-defined minimum local whitespace (`minLocalWS`), partitioning will define a tentative cutline that divides the bin's placement area in half. Partitioning targets an equal division of cell area, but is given more freedom to deviate from its target. Tolerance is computed so that with whitespace degradation, each descendant bin of the current bin will have at least `minLocalWS`. After a partitionment is calculated, the cutline is shifted to ensure that `minLocalWS` is preserved on both sides of the cutline. If the minimum local whitespace is chosen to be small, Capo can produce tightly packed placements which greatly improves wirelength.

Safe Whitespace. The last whitespace allocation mode is designed for bins with "large" quantities of whitespace. In safe whitespace allocation, as with minimum local whitespace allocation, a tentative geometric cutline of the bin is chosen, and the target of partitioning is an equal bisection of the cell area. The difference in safe whitespace allocation mode is that the partitioning tolerance is much higher. Essentially, any partitioning solution that leaves at least `safeWS` on either side of the cutline is considered legal. This allows for very tight packing and reduces wirelength, but is not recommended for congestion-driven placement.

Figure 1 illustrates Capo's uniform and non-uniform whitespace allocation. Column (a) shows global placements with uniform (top) and non-uniform (bottom) whitespace allocation on the ISPD 2005 contest benchmark adaptec1 (57.34% utlization). In the non-uniform placement shown, the minimum local whitespace is 12% and safe whitespace is 14%. Columns (b) and (c) show intensity maps of the local utilization of each placement. Lighter areas of the inten-

sity maps signify violations of a given target placement density; darker areas have utilization below the target. Regions completely occupied by fixed obstacles are shaded as if they exactly meet the target density. The target densities for columns (b) and (c) are 90% and 60%. Note that uniform whitespace produces almost no violations when the target is 90% and relatively few when the target is 60%. The non-uniform placement has more violations as compared to the uniform placement especially when the target is 60%, but remains largely legal with the 90% target density.

3. WHITESPACE IN DETAIL PLACEMENT

Capo uses several different techniques to further reduce HPWL after global placement such as the sliding window optimizer RowIroning and a greedy cell movement scheme described below. In addition, Capo 10 performs optimal whitespace allocation using linear programming without changing relative cell ordering [3, 13].

RowIroning. In RowIroning, optimal placers based on branch-and-bound and dynamic programming techniques replace windows of cells and whitespace chosen from the placement area [4]. These placers pack cells, and whitespace is represented by fake cells. To model whitespace accurately, one fake cell per site is needed, but Capo evenly divides contiguous regions of whitespace into at most three fake cells to limit runtime. This window of local improvement moves over all cells in left-to-right and top-to-bottom order (or the opposite directions). Because the optimizations are local, they rarely affect overall whitespace distribution.

Greedy Cell Movement. Capo makes use of a gridded greedy movement technique to improve both wirelength and whitespace distribution. A grid is imposed on the placement region to analyze local placement density. For cells that are in regions with density violations, candidate legal new locations are found in areas of lower density violation. Candidate moves are ranked by how well they alleviate the violations and how they affect wirelength. Moves are made until a threshold of improvement is reached. We have found this to be a fast and effective method of removing density violations without adversely affecting wirelength.

4. WHITESPACE AND ROUTABILITY

With uniform whitespace allocation, Capo typically produces routable placements, but some congested areas remain. Capo 10 implements a whitespace allocation scheme described in [11] to improve placement routability. This technique uses a congestion map to estimate routing congestion after each layer of min-cut placement. Based on the congestion estimates, whitespace is allocated preferentially to areas of high congestion through cutline shifting. Coupled with other techniques from ROOSTER [11], Capo 10 outperforms best published routed wirelengths and via counts.

5. IMPROVED MIXED-SIZE PLACEMENT

Industrial floorplacement problems are increasingly difficult due to factors such as an increasing number of movable modules and a wide variation of module sizes. There is also insufficient cohesion for whitespace allocation between top-down methods and macro-placement algorithms. For example, a partitioner may misapproximate the area required by a set of macros and incorrectly allocate whitespace. To address these issues, we have integrated into Capo 10 the SCAMPI (SCalable Advanced Macro Placement Improvements) work [9]. The top-down partitioning flow is modified to selectively place large macros, while smaller macros are clustered into soft modules that will be placed later. The robustness of the flow is also improved by employing fast *look-ahead* Simulated Annealing on large macros of newly created bins. This allows early detection of bins difficult to floorplan, and alerts the placer to backtrack and seek a different partitioning solution.

Commandline options	Functions
GENERIC OPTIONS	
`-f filename.aux`	Read the input from `filename.aux` (LEF/DEF or Bookshelf formats).
`-num n`	Run Capo n times and choose the best.
`-save`	Save the placement in Bookshelf or LEF/DEF format (depending on input).
`-plot filename`	Produce an image of the placement as a GNUplot [14] script named `filename.plt`.
`-noCapo`	Run detail placement only.
META OPTIONS	
`-ECO`	Have Capo legalize and optimize a given initial placement.
`-ROOSTER`	Improve routability. See Section 4.
`-SCAMPI`	Improve mixed-size placement. See Section 5.
`-ispd06 0-100%`	Enforce the rules of the ISPD 2006 Placement contest and set a target placement density.
`-faster`	Trade solution quality for speed by disabling placement feedback [8] and weighted partitioning [6, 12].
`-tryHarder`	Trade runtime for solution quality by increasing calls to partitioners and enabling placement feedback [8] and weighted partitioning [6, 12].
`-saveMem On,Auto,Off`	"`On`": enable memory saving options that may increase runtime. "`Auto`" (default): enable memory saving options based on benchmark size and available system memory.
WHITESPACE OPTIONS	
`-safeWS 0-100%`	Enable non-uniform whitespace allocation. See Section 2.
`-minLocalWS 0-100%`	Enable non-uniform whitespace allocation. See Section 2.
`-uniformWS`	Enable uniform whitespace allocation. See Section 2.

Table 1: Frequently used options in MetaPlacer/Capo 10.

6. WHITESPACE AND MEMORY PROFILE

Capo's non-uniform whitespace allocation techniques tend to produce unbalanced partitionments at the top layers. As peak memory usage grows with partitioning problem size, memory consumption can stay near the peak for longer periods of time during placement. To counteract the increased possibility of thrashing, Capo 10 has several memory improvements which include the slimming down of data structures and carefully choosing the lifetimes of major data structures so that fewer need to be in main memory simultaneously. The most radical of these changes involves removing the netlist hypergraph from main memory during the largest partitioning instances and rebuilding it from scratch afterwards. These changes reduce peak memory consumption by 2x compared to Capo 9.1 but slow down global placement by 10%.

Acknowledgments

This work was partially supported by the DARPA/MARCO Gigascale Systems Research Center, the National Science Foundation, Synplicity, as well as equipment donations from Intel.

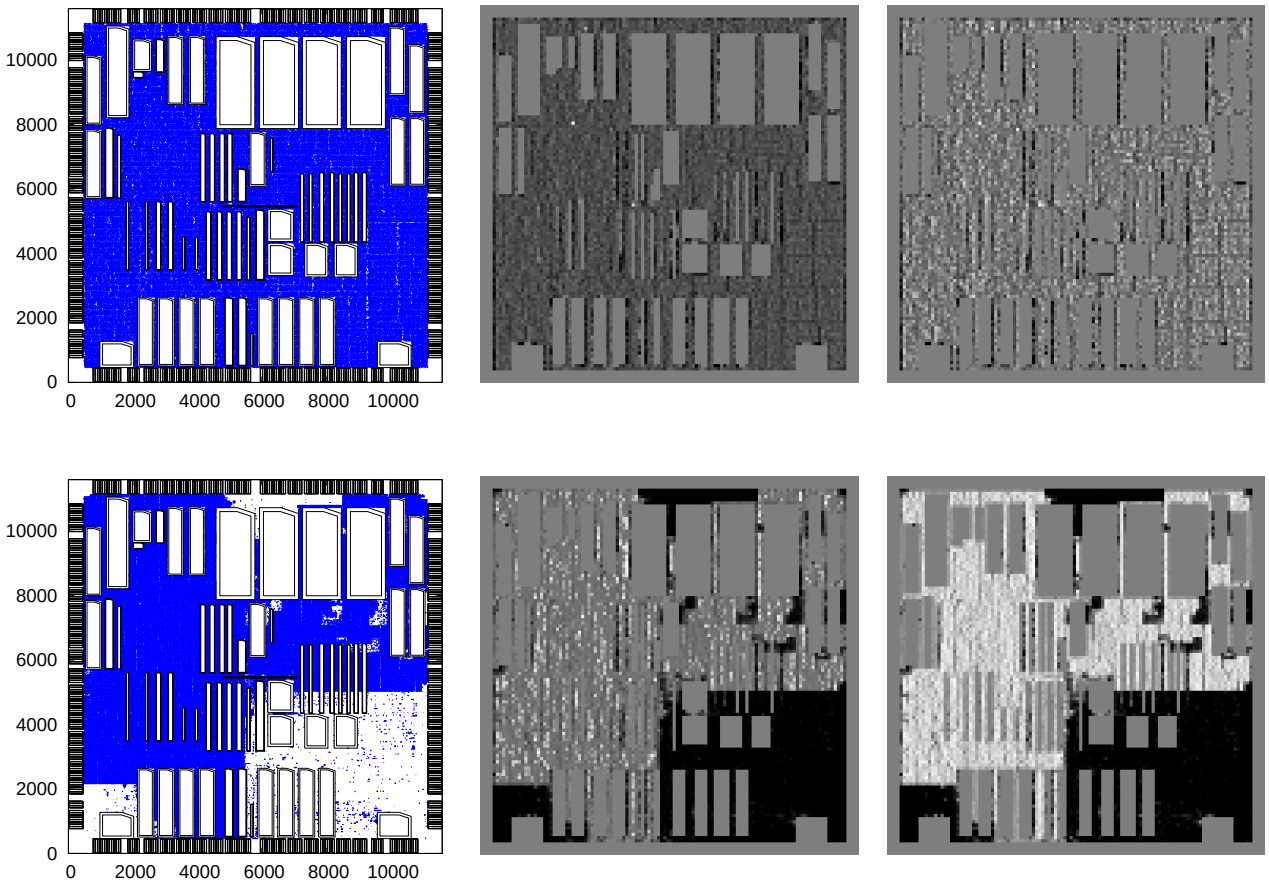

Figure 1: Column (a) shows Capo 10 global placements of the contest benchmark adaptec1 with uniform whitespace allocation (top) and non-uniform whitespace allocation (bottom). Fixed obstacles are drawn with double lines. To indicate orientation, north-west corners of blocks are truncated. Columns (b) and (c) depict the local utilization of the uniform and non-uniform placements. Lighter areas of the placement signify regions that violate the target placement density whereas darker areas have utilization below the target. Areas with no placeable area (such as those with fixed obstacles) are shaded as if they exactly meet the target density. The target placement density for column (b) is 90% and the target for column (c) is 60%. Adaptec1 has 57.34% utilization. The HPWL for the uniform and non-uniform placements is 10.6924e7 and 9.032e7 respectively. As the intensity maps show, when 60% utilization is the target, uniform whitespace allocation is much more appropriate than 12% minimum local whitespace. On the other hand, 12% minimum local whitespace is appropriate in terms of violations when the target is 90% utilization and has much better wirelength.

7. REFERENCES

[1] S. N. Adya, I. L. Markov, P. G. Villarrubia, "On Whitespace and Stability in Mixed-size Placement and Physical Synthesis," *ICCAD* 2003, pp. 311-318.

[2] C. J. Alpert, G.-J. Nam, P. G. Villarrubia, "Free-Space Management for Cut-Based Placement," *ICCAD* 2002, p. 746.

[3] U. Brenner and J. Vygen, "Faster Optimal Single-Row Placement with Fixed Ordering," *DATE* 2000, pp. 117-121.

[4] A. E. Caldwell, A. B. Kahng, I. L. Markov, "Optimal Partitioners and End-case Placers for Standard-cell Layout," *IEEE Trans. on CAD* 19(11), pp. 1304-1314, 2000.

[5] A. E. Caldwell, A. B. Kahng, I. L. Markov, "Hierarchical Whitespace Allocation in Top-down Placement," *IEEE Transactions on CAD* 22(11), Nov, 2003, pp. 716-724.

[6] T. C. Chen, Y. W. Chang and S. C. Lin, "IMF: Interconnect-Driven Multilevel Floorplanning for Large-Scale Building-Module Designs," *ICCAD*, pp. 159-164, November 2005.

[7] J. Cong, M. Romesis and J. Shinnerl, "Robust Mixed-Size Placement Under Tight White-Space Constraints," *ICCAD* 2005.

[8] A. B. Kahng and S. Reda, "Placement Feedback: A Concept and Method for Better Min-cut Placement," *DAC* 2004, pp. 357-362.

[9] A. N. Ng, R. Aggarwal, V. Ramachandran and I. L. Markov "Solving Hard Instances of Floorplacement", to appear in *Proc. Int'l Symp. on Physical Design (ISPD)* , San Jose, CA, 2006.

[10] J. A. Roy, S. N. Adya, D. A. Papa and I. L. Markov, "Min-cut Floorplacement," to appear *IEEE Trans. on CAD*, 2006.

[11] J. A. Roy, J. F. Lu and I. L. Markov, "Seeing the Forest and the Trees: Steiner Wirelength Optimization in Placement", to appear in *Proc. Int'l Symp. on Physical Design (ISPD)* , San Jose, CA, 2006.

[12] N. Selvakkumaran and G. Karypis, "THETO: A Fast and High-Quality Partitioning Driven Global Placer," Technical Report 03-046, 2003, University of Minnesota.

[13] X. Tang, R. Tian, M. D.F. Wong, "Optimal Redistribution of White Space for Wire Length Minimization," *ASPDAC* 2005, p. 412.

[14] `http://www.gnuplot.info`