Classical Floorplanning Harmful?*

Andrew B. Kahng
UCLA Computer Science Department
3713 Boelter Hall
Los Angeles, CA 90095-1596 USA
abk@cs.ucla.edu

ABSTRACT

Classical floorplanning formulations may lead researchers to solve the wrong problems. This paper points out several examples, including (i) the preoccupation with packing-driven, as opposed to connectivity-driven, problem formulations and benchmarking standards; (ii) the preoccupation with rectangular (and L or T shaped) block shapes; and (iii) the lack of attention to algorithm scalability, fixed-die layout requirements, and the overall RTL-down methodology context. The right problem formulations must match the purpose and context of prevailing RTL-down design methodologies, and must be neither overconstrained nor underconstrained. The right solution ingredients are those which are scalable while delivering good solution quality according to relevant metrics. We also describe new problem formulations and solution ingredients, notably a *perfect rectilinear floorplanning* formulation that seeks zero-whitespace, perfectly packed rectilinear floorplans in a fixed-die regime. The paper closes with a list of questions for future research.

Categories and Subject Descriptors

B.7.2 [**Hardware**]: Integrated Circuits— *design aids; placement and routing*; F.2.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity—*geometrical problems and computations; routing and layout*

General Terms

VLSI floorplanning, coarse placement, block packing and layout, hierarchical design methodology.

1. INTRODUCTION

Classical floorplanning takes as input a set of blocks, a netlist, and a target layout region. The (rectangular) blocks may be *hard*, *soft* or *semi-soft*. Hard blocks have fixed aspect ratio and pin locations. Soft blocks have fixed area, with (upper- and lower-bounded) continuously variable aspect ratio. Semi-soft blocks have fixed area with discrete allowed aspect ratio, usually corresponding to alternative block realizations (e.g., allowed foldings of datapaths).[1] Since the mid-1990s, L- and T-shaped block shapes have been specifiable and/or permissible; this is motivated by enforced colocation of a datapath block with related control, or the shape of particular types of datapath blocks. The target layout region is typically specified by upper and lower bounds on its aspect ratio, e.g., a square layout has upper bound = lower bound = 1.

The classical floorplanning problem seeks to shape and pack all blocks, such that no blocks overlap and the enclosing layout region has minimum area while satisfying aspect ratio constraints. This corresponds to a *minimum whitespace* objective. Optionally, the packing can attempt to minimize an estimate of the wiring needed to realize the netlist connectivity. This corresponds to a *minimum wirelength* objective.[2]

With respect to wiring, early works such as [13] focused on channeled block layouts and the associated global routing and pin assignment issues. Today, for various reasons – fixed-die regime, presynthesis floorplanning, N-layer metal with available over-the-block routing – channelless layouts are the norm and the top-level routing is more or less viewed (or more accurately, ignored) as "area-routed".

This invited paper addresses the question, "Classical floorplanning harmful?" The "classical floorplanning" problem has inspired an immense literature of valuable research results. Nevertheless, "classical floorplanning" has in several ways led researchers and practitioners to focus on the wrong problem. Examples include (i) preoccupation with packing driven, as opposed to connectivity driven, problem formulations and benchmarking practices; (ii) preoccupation with only rectangular (and L- or T-shaped) block shapes; (iii) lack of attention to the fixed-die context; (iv) lack of attention to the overall RTL-down methodology context, and to whether there are any real differences between "floorplanning" and "hierarchical approaches to achieving placement"; and (v) lack of attention to achieving a holistic, scalable approach. The main conclusion is that *the problem must be changed.* The right problem formulations must match the purpose of floorplanning (i.e., coarse placement enabling

*This research supported by Cadence Design Systems, Inc. and the MARCO Gigascale Silicon Research Center.

[1] An alternative taxonomy lumps soft blocks together with semi-soft blocks: since the layout region is row-based, block heights are discrete and there is no such thing as a block with continuously variable aspect ratio.

[2] The minimum wirelength objective is not yet consistently addressed in either the definition or the reporting of benchmarks in the literature.

route planning), and the context of today's convergent RTL-down design methodologies. Furthermore, the right formulations must be neither overconstrained nor underconstrained. The right solution ingredients are those which are scalable while delivering good solution quality according to relevant metrics.

The remainder of this paper is organized as follows. Section 2 of this paper gives limitations of the classical floorplanning formulation, along with desiderata for an improved formulation. Section 3 sketches a "perfect rectilinear floorplanning problem" (PRFP) formulation, along with an example solution flow and supporting algorithm technologies. The PRFP seeks zero-whitespace, perfectly packed rectilinear floorplans in a fixed-die regime, and is presented as an example alternative to classical floorplanning. Section 4 concludes with a few open research questions.

2. LIMITATIONS OF CLASSICAL FLOORPLANNING

A modern floorplanner must deal with (i) up to thousands of blocks, including hard IP blocks and presynthesis RTL soft blocks; (ii) timing and routability objectives, as well as tight interactions with route planning and performance optimization tools; and (iii) a fixed-die layout resource with discreteness (cell row pitch) in the vertical dimension. (This is not very different from what a placer deals with!) In this section, we compare these modern floorplanning requirements against the "classical floorplanning" formulation.

2.1 Obsession With Packing

The classical floorplanning literature has had an obsession with *packing* (an NP-hard problem) and the minimum-whitespace objective.[3] Floorplan representations have evolved from dual graphs (the standard representation in literatures such as facility layout and architecture since the early 1970s [17] [8]) to slicing trees, to sequence-pairs [37], bounded-slicing grids [38], O-trees [18], etc. This evolution has been driven by the quest for a complete and irredundant representation that allows efficient heuristic search for *good packings*.[4] Since the solution spaces are so large, appropriate neighborhood operators for iterative optimization metaheuristics (e.g., simulated annealing, evolutionary optimization, or tabu search) are also sought. We make the following observations.

- Recent "complete" representations (SP, BSG, O-tree) require heavy coercion to deal with natural instance attributes (e.g., fixed blocks, alignment constraints, and pitch-matching constraints for datapaths). Furthermore, they suffer from combinatorial explosion when the instance contains semi-soft blocks that have many alternative realizations. *Scalability* to instances of hundreds or thousands of blocks seems out of reach.

- Working with a floorplan *representation*, rather than directly with a floorplan *realization*, has overheads in terms of cost function evaluation and accuracy. For example, sizing and compaction steps are required before one can even estimate the wirelength corresponding to a given sequence-pair.

- Complex objectives such as path-timing or wirelength seem difficult to handle with, e.g., annealing of sequence pairs. Notice, for example, that the whitespace and wirelength objectives are not simultaneously optimized by floorplan compaction.

- To handle objectives such as wirelength and path timing, one must focus on connectivity. Indeed, the correct approach must be *connectivity-centric*, rather than *packing-centric*. Packing must be made into a secondary issue – or ideally, a non-issue.

2.2 Design Methodology Context

Floorplanning coarsely maps portions of the IC design to portions of the IC layout region. It delivers "coarse embedding" or "coarse placement" to support global interconnect planning and performance optimizations. We observe:

- (Hierarchical) floorplanning is the natural response to three facts of life: (i) complexity, (ii) the need for forward prediction in a convergent design process, and (iii) the fact that the only useful prediction technologies we know of are constructive predictors. When upstream tools cannot predict downstream outcomes, they must constructively (e.g., by floorplanning and top-level route planning) predict, then constrain, the downstream outcomes. Hierarchy arises because humans are limited in how many things they can think of at once. Hence, the sequence of no-floorplanning, physical-floorplanning, RTL-floorplanning, etc. is natural as design complexities increase.

- To find the most useful floorplanning formulations, especially in light of "hierarchical floorplanning is hierarchical coarse placement", we should ask whether there are any *fundamental* differences between the purpose of floorplanning and the purpose of placement.[5] Along a similar vein, we should ask whether the design process is really different depending on whether hierarchy management is automated (within a block-based hierarchical floorplanning tool) or manual. If the answer to either question is "no", then floorplanners should probably look more like placers than packers.[6]

- In modern methodologies, RTL is partitioned, floorplanned, route-planned and optimized for performance before logic synthesis and place-and-route. Hence, area and performance are at best crudely estimated (hopefully within 15%, but sensitive to block aspect ratios, actual critical path structure after top-level routing and pin assignment, etc.). The amount of effort spent in addressing the resulting floorplan instances – and in

[3]Published benchmarking practices and floorplanning metrics all focus on whitespace. We note that a more pragmatic reaction to an NP-hard problem is to avoid it as much as possible.

[4]Other, more greedy/constructive approaches such as shelf-packing or cluster growth in combination with zone-refinement [28] have been attempted. However, solution quality is poor (and scales even more poorly).

[5]For example, non-rectangular shapes and noisy area estimates occur in floorplanning, but not in placement. However, these aspects of floorplanning instances may be *side effects* of current abstractions of RT-level (timing and chip planning) design needs.

[6]Advances in design and process technologies could help remove distinctions between floorplanning and placement. For example, better understanding of delay sensitivity with respect to sizing, and repeater library design, can together eliminate the need for buffer block methodologies. A second active device layer can make repeater insertion and associated floorplanning issues trivial [47].

particular, their "zero whitespace" packing objectives – must be impedance-matched to this level of accuracy.[7]

2.3 Overconstrained Shaping

Soft floorplan blocks are created by partitioning and reclustering of RT-level HDL code. Each block contains hundreds or thousands of standard cells, and is hence *very granular.* The granularity of the layout resource is even finer: cell row pitch (a few routing tracks) in the vertical dimension, and site width (less than a single track) in the horizontal dimension. We make several observations.

- Restricting pre-synthesis or pre-place-and-route soft blocks to rectangular, L- or T- shapes is an overconstraint. Even if hard blocks and synthesized datapaths with regular structure are present in the netlist, most blocks will be amenable to non-rectangular shaping.[8]

- Even though "round" blocks with low aspect ratio help wire estimation, this does not mean that block shapes can have only four, six or eight sides (rectangle, L, T respectively). Roundness of a polygonal shape, in fact, has very little to do with the number of sides.

- No requirement for particular block shapes arises from the downstream place-and-route tools. For example, irregular block outlines can be handled by balance constraints in a partitioning-based approach (similar to how PDEF3.0 region constraints are handled in top-down placers today). Analytic placers can also impose region constraints at appropriate levels of granularity to reflect complex block shapes.

- Finally, it is not even necessary for the blocks to be non-overlapping as long as there is sufficient cell area assigned to all blocks. As noted in the previous footnote, arguably blocks *should* overlap and intermingle their contents. One possible formulation is to allow any given hierarchy block to cover its footprint with varying "depths" (i.e., densities with respect to available sites). Then, a legal floorplan solution is one that has total depth = 1 everywhere.[9]

[7]In addition, the modern use model for floorplanning entails partial and probabilistic information, as well as many types of ECOs. Thus, an ideal framework should deliver *incremental* and *anytime* [6] solutions.

[8]Color plots of logic hierarchy vs. final placed location in highly optimized, flat placements show that cells of hierarchy blocks tend to intermingle at the block boundaries. Accordingly, terms like "amoeba placement", "flexible block", etc. abound in the commercial EDA world today. For logic that is not built into a given system-on-chip platform, hard-IP reuse is likely to decrease because of rapid scaling and divergent process recipes. Thus, there will be more "softness" in the part of the design that is synthesized from RTL to layout.

[9]For example, two blocks with equal amounts of cell area could be placed into adjacent disjoint regions, with each block having depth = 1 in its respective region. An alternative would be to place each block with uniform depth = 1/2 into the *union* of the two regions. This idea was first proposed in 1987 by Prof. T. C. Hu in the context of a "TACP" (tentative assignment and competitive pricing) approach to placement.

2.4 Underconstrained Layout Region

The classical floorplanning literature treats the die area as having constrained aspect ratio, but unbounded size. The objective is to "find the packing with smallest containing die". In reality, the use of floorplanning during the chip synthesis process almost always comes *after* the die size and package have been chosen. Thus, floorplanning should be cast as a *fixed-die* problem, and the packing *must simultaneously achieve zero whitespace and zero overlap* for the given choice of fixed die. Current formulations do not enforce this constraint.

3. PERFECT RECTILINEAR FLOORPLANNING

In this section, we describe a new *perfect rectilinear floorplanning problem* (PRFP) formulation that addresses many of the above issues. The discussion draws largely on the presentation in [9]. The PRFP formulation entails (i) a fixed-die, zero-whitespace layout region into which *rectilinear* blocks must be perfectly packed; (ii) discreteness constraints on vertical edges in block boundaries (corresponding to row pitch in standard-cell layouts); and (iii) *rectilinear* blocks that may be continuously shaped while maintaining fixed area.

Our premise is that no matter how good the block packer is, there will be whitespace and overlaps in its solution. PRFP welcomes such "global floorplanning solutions", and turns them into zero-whitespace, zero-overlap "detailed floorplanning solutions" with minimum perturbation.

3.1 An Example PRFP-Based Flow

An example flow built around the PRFP formulation might be as follows.

1. *RTL partitioning* to create blocks. Hierarchy-aware, bus structure-aware aware, timing constraint-aware recursive 2- and k-way partitioning engines can be built from existing algorithm components in the literature. An important open research issue addresses the interaction between the block definition process (i.e., the partitioning objective and constraints) and the floorplan quality that is achievable with the resulting set of blocks.

2. *Global floorplanning* to create a connectivity- and timing-driven, near-legal floorplan. The result of global floorplanning can have whitespace and overlaps. We believe that the key technologies for global floorplanning will be based on recursive top-down partitioning, since this affords a spatially convergent creation of the embedding. At the top levels of the physical hierarchy, packing effects are negligible. At the bottom levels (e.g., < 10 blocks), branch-and-bound end case shaping/packing can be applied to packing representations such as O-trees.

3. *Detailed floorplanning* to obtain a set of rectilinear block shapes with zero-whitespace and zero-overlap. The detailed floorplanning solution should be as close as possible to the global floorplanning solution, and the complexity of block outlines should be minimized (see the discussion of shape metrics, below). Top-level routing and pin assignment can be performed in iteration with both global and detailed floorplanning.

4. *Generic standard-cell placement* to achieve a legal, timing- and routability-driven gate-level placement that respects detailed floorplan block outlines (if not as region constraints, then as "suggestions").

This PRFP-based flow doubtless resembles existing RTL-down methodologies. However, it is more direct in exploiting the flexibility and high quality of standard-cell placement (which typically follows block placement and logic synthesis). It is consistent with flows in which standard-cell block synthesis and place-and-route follow after (soft-block dominated) RTL floorplanning (i.e., PRFP serves either as a cleanup step after traditional floorplanning, or as the end-case optimizer for recursive partitioning-based floorplanning). The logic synthesis and timing optimizations occur in "coarse mode" at the beginning of Step (3), and in "detailed mode" at the beginning of Step (4). There is iteration. PRFP also removes unnecessary and artificial constraints on the floorplan solution that stem from the traditional "block packing" perspective; it reduces the pressure on block packing algorithms and allows more connection-centric (rather than packing-centric) approaches to be used.

3.2 Some Details of Detailed Floorplanning

We now sketch some thoughts on the PRFP-based detailed floorplanning part of the above flow. We are given n rectangles $R_1, \ldots, R_n$ (some of them are overlapping) in the layout region, as illustrated in Figure 1(a).[10] The layout region is a "global" rectangle with fixed width W and height H. We have that the sum of the areas of rectangles is equal to the area of the layout region ($W \cdot H$).

Turning the global floorplan into a detailed floorplan essentially means that overlaps must be migrated toward whitespace, and vice-versa, until both disappear. This area migration (legalization) entails the following subproblems.

P1. Find all intersecting pairs of rectangles (overlapping or touching each other); and

P2. Find all white simple rectilinear polygons (white spaces that exist in the layout region because of overlapping rectangles) and their areas.

Subproblem P1 can be solved in $O(n^2)$ time directly, or optimally in $O(nlogn+s)$ time (where s is the number of intersecting pairs) by using plane-sweep techniques supported by interval trees (see pp. 359-363 in [1]). Subproblem P2 can be solved as follows (see pp. 13-15 and 340-347 in [1]). Using a plane-sweep technique supported by segment trees, determine the contour of the union F of n rectangles $R_1, \ldots, R_n$, i.e., a collection of disjoint cycles composed of (alternating) vertical and horizontal edges. This is illustrated in Figure 1(b).

By convention, any edge is directed in such a way that we have the figure on the left while traversing the edge. This is equivalent to saying that a cycle is oriented clockwise if it is the boundary of a hole (i.e., an inner whitespace), and counterclockwise if it is an external boundary of a connected component. Since all rectangles are in the layout region (global rectangle of size $W * H$), external boundary cycles are located inside of the global $W * H$-rectangle. So, the holes and the regions between the boundary of the global $W*H$-rectangle and the external boundary cycles comprise our simple rectilinear polygons of whitespace. (See Figure 1(c).) The time complexity of this approach is $O(nlogn + p\ log(n^2/p))$, where p is the number of edges in the contour that we find. Note that here we can have some flexibility to shift the global $W*H$-rectangle horizontally within a range of $W - (x_{max} - x_{min})$ units and vertically within a range of $H - (y_{max} - y_{min})$ units (here, x_{min} is the leftmost x-coordinate of the vertices of rectangles, x_{max} is the rightmost, etc.). By slightly moving the global $W*H$-rectangle, we may improve the distribution of white spaces with respect to the distribution of the overlapping areas. Finally, the area of a simple rectilinear polygon with k vertices can be computed in $O(k)$ time (e.g., [10]).

Given all this information, we can build a graph $G = (V, E)$ (see Figure 1(d)):

- With each rectangle associate a blue node (so, $p_1, ..., p_n$);
- With each simple white subpolygon, a white node (so, $w_1, ..., w_m$);
- Draw an edge between p_i and w_j iff these polygons share a common boundary;
- Draw an edge (p_i, p_j) iff R_i and R_j share a common boundary but do not have a common inner point;
- Between each R_i and R_j which share a common inner point (i.e., intersect each other) draw an edge (p_i, p_j). Furthermore, associate with this intersection an additional red vertex s_{ij} and draw two edges (s_{ij}, p_i) and (s_{ij}, p_j).

Using this graph, we can find efficient ways to migrate area from overlap-spaces to white-spaces. A resulting iterative greedy heuristic for area migration (legalization) might be as follows.[11]

while there is a white subpolygon
  do choose a white subpolygon w of largest area
    using the graph G find the closest to w red vertex r
    migrate area of size $x = min\{area(w), area(r)\}$
      from r to w along a shortest (r, w)-path
    put $area(r) = area(r) - x$, $area(w) = area(w) - x$
    if $area(r) = 0$ **then** delete r from G
    if $area(w) = 0$ **then** delete w from G
    update the graph G

At each iteration of the algorithm, we obtain a migration path and an amount of area to migrate from an overlap-space to a white-space. For example, for an edge (v, u) of the (r, w)-path connecting two blue vertices or a blue vertex with a white vertex, x gives us the area that must migrate from polygon p_v to polygon p_u (polygon p_v borrows an area of size x from polygon p_u). For details, see [9]. Note that migrating area of size x from an overlap-space to a white-space

[10] The input does not have to consist of rectangular block shapes. We use this for simplicity, and to emphasize that detailed floorplanning can be applied to results of existing block packers.

[11] Other approaches are discussed in [9]: flow and/or transportation approaches can operate on a structure similar to the dual graph of the floorplan; physical-analog systems can be simulated; and geometric matching techniques can also be used to match up overlap with nearby whitespace.

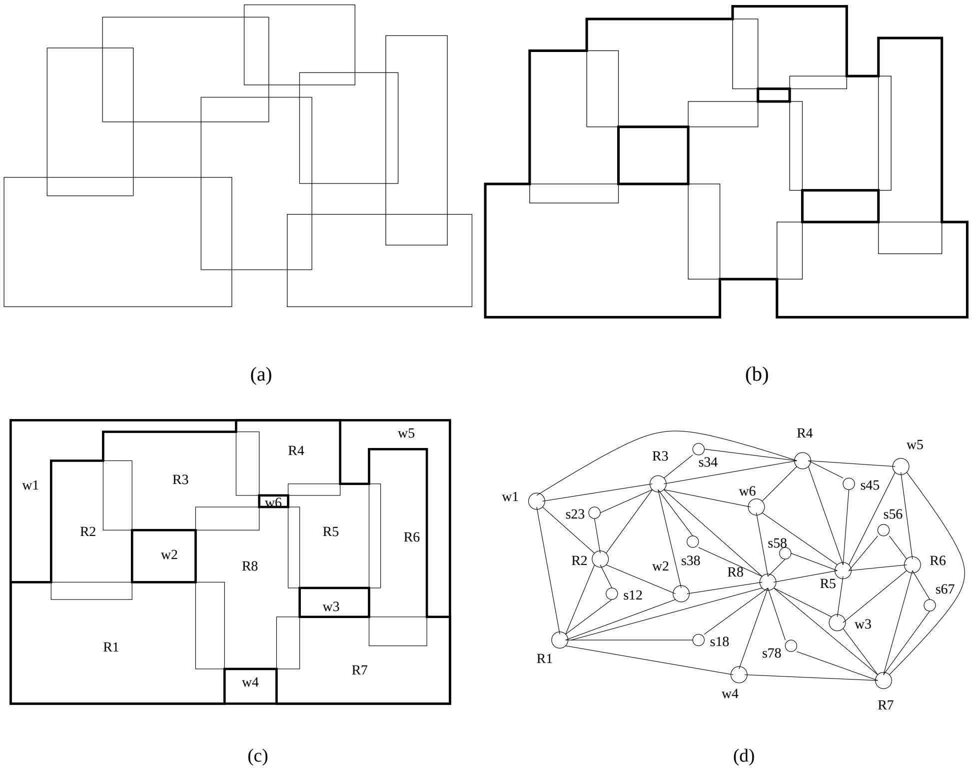

Figure 1: (a) A global floorplanning result (rectangles); (b) the contour (defining whitespace); (c) whitespace and overlap within the layout region; and (d) the corresponding graph.

can change the adjacency between vertices in the graph G (e.g., p_i and p_j are no longer neighbors, or they become neighbors). So, after each iteration our graph can change and we must update G.

The above approach (and other intuitive heuristics) generally entail solving the following subproblems many times over.

P3. For two given simple rectilinear polygons P_i, P_j, that touch each other, find their common boundary Γ_{ij} - a collection of disjoint chains composed of (alternating) vertical and horizontal edges belonging to boundaries of both polygons;

P4. Find a way to simplify the shape of Γ_{ij} while transferring an amount of area x from P_j to P_i (without disconnecting either of these polygons).

Subproblem P3 can be directly solved in $O((|P_i|+|P_j|)log|P_i|)$ time. Subproblem P4 is the key to delivering a reasonable detailed floorplanning result via area migration: it enables chains of “area borrowing” between regions. Addressing P4 first requires a formal objective that captures “simplicity” of a shape. To this end, we know that for a fixed area, the perimeter of an object increases as it becomes more irregular in shape. Hence, we can use the perimeter of a polygon as a measure of its shape irregularity. (For example, the perimeter of a rectilinear object is minimized if the object is square-shaped.) Subproblem P4 is thus formalized as [9]: Given two simple rectilinear polygons P_1 and P_2 that touch each other, and their common boundary Γ_{12}. We assume

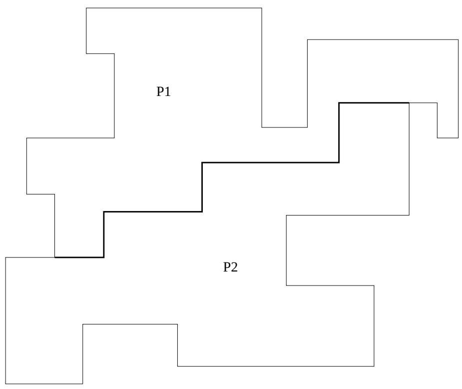

Figure 2: Two polygons with common boundary.

that Γ_{12} is connected, i.e., it is a chain of (alternating) vertical and horizontal edges belonging to boundaries of both polygons (see Figure 2). If Γ_{12} is not connected, then we can work on each connected part separately.

We seek to redraw the common boundary Γ_{12} so as to define two new polygons P_1' and P_2' in such a way that the length of the new common boundary Γ_{12}' is minimized, subject to

- both P_1' and P_2' are simple rectilinear polygons;
- $P_1 \cup P_2$ and $P_1' \cup P_2'$ are equal;
- $area(P_1) = area(P_1') + x$, $area(P_2) = area(P_2') - x$, where x is a given value; and

- $\frac{area(P_1 \cap P_1')}{area(P_1)} \geq c$, where c is a given constant.

Although this problem is NP-hard [9], it appears to be amenable to effective heuristics within a PRFP-based flow. Notice that the variant of the problem when $x = 0$ is also of interest since we may want to improve the boundary between two polygons without borrowing any area (simply to make the shapes of polygons more regular); see also [24].

4. CONCLUSIONS AND OPEN QUESTIONS

In this paper, I have tried to marshal support for a somewhat extreme and provocative title, "Classical Floorplanning Harmful?". Of course, classical floorplanning has provided great benefits to researchers and designers alike, and any attempt to box its rich literature into a "strawman" is bound to displease some. Nevertheless, there are indeed several ways in which "classical floorplanning formulations" have diverged from real-world physical chip implementation requirements. The PRFP-based flow shows that it is possible to remain purely connectivity- and timing-driven until very late in the "floorplanning" game, and that packing *can* be made into a virtual non-issue. The question of whether this is still floorplanning, or simply a retargeting of top-down placement, is more than a matter of semantics: it highlights the need to identify unique and differentiating aspects of *floorplanning* within today's convergent, performance-driven spatial embedding methodologies.[12]

I will conclude with a few research directions.

- *Flexibility metrics.* Intuitively, *flexibility* in a floorplanning instance should capture the amount of correlation between minimum-wirelength solutions and minimum-whitespace solutions. Given the same block netlist and areas, a more flexible variant of the floorplanning instance (e.g., with more allowed shapes for each block) should be packable with less whitespace than a less flexible variant, while achieving the same or better wirelength. Alternatively, flexibility indicates the length scale at which localization starts to interact with packing. Interesting metrics have been proposed in [5] and [45], but finding a good metric with strong empirical validation is still open.

- *Shape metrics.* As detailed floorplanning turns a global floorplan into a perfect, zero-whitespace, zero-overlap solution, it should change the initial solution as little as possible. Thus, one possible objective function is the sum of *shape distances* between the original and final shapes. Since two-dimensional shapes are specified by the planar curves forming their boundaries, it is natural to seek a formal measure of how similar two given curves are to each other. There is a rich literature, encompassing the Hausdorff metric [2], the Frechet metric [3], similarity measures that are invariant under similitude transformations [20] [4], Fourier descriptors [26] [33], tree matching [35], etc. However, none of these metrics satisfies even simple desiderata for our PRFP context. An ideal shape metric should distinguish between changes of aspect ratio, penalize complex shapes, encourage blocks to remain as close to their original locations as possible, and be quickly computable [9].

- *Partitioning objectives that yield friendly floorplanning instances.* As noted above, no research has yet addressed the interaction between the block definition process (i.e., the RTL partitioning objective and constraints) and the floorplan quality that is achievable with the resulting set of blocks. However, this issue must be understood for the best solutions to remain attainable as far down into the design process as possible.

[12]The methods proposed for achieving global floorplans are essentially placement methods: recursive partitioning, and/or analytic placement with appropriate spreading. These appear to have better scalability than iterative-search methods, and better solution quality than greedy methods such as cluster-growth or certain zone-refinement variants.

5. ACKNOWLEDGMENTS

This paper draws heavily on recent results obtained with Andy Caldwell and Dr. Feodor Dragan at UCLA, cited as [9]. Discussions with Doug Carroll (on flexibility metrics) and with Dr. Ravi Varadarajan (on RTL-down planning in general) have also been invaluable.

6. REFERENCES

[1] R.K. Ahuja, T. L. Magnanti and J.B. Orlin, *Network Flows*, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[2] H. Alt, B. Behrends and J. Blömer, "Approximate Matching of Polygonal Shapes", *Proc. ACM Symp. on Computational Geometry*, 1991, pp. 186-193.

[3] H. Alt and M. Godau, "Measuring the Resemblance of Polygonal Curves", *8th Annual Symp. on Computational Geometry*, 1992, pp. 102-109.

[4] E.M. Arkin, L.P. Chew, D.P. Huttenlocher, K. Kedem and J.S.B. Mitchell, "An Efficiently Computable Metric for Comparing Polygonal Shapes", *IEEE Trans. on PAMI* 13 (1991), pp. 209-216.

[5] K. Bazargan, S. Kim and M. Sarrafzadeh, "Nostradamus: A Floorplanner of Uncertain Designs", *Proc. ISPD*, 1998, pp. 18-23.

[6] M. Boddy and T. Dean, "Solving Time-Dependent Planning Problems", in *Proc. IJCAI*, 1989, pp. 979-984.

[7] H. Botatia, "A Projective Invariant Metric for Measurement of Similarity Between Two Polygons". *Proceedings of Europe - China Workshop on Geometrical Modeling and Invariants for Computer Vision*, 1995, pp. 307-314.

[8] Y. A. Bozer, R. D. Meller and S. J. Erlebacher, "An Improvement-Type Layout Algorithm for Single and Multiple-Floor Facilities", *Management Science* 40(7) (1994), pp. 918-932.

[9] A. E. Caldwell, F. Dragan and A. B. Kahng, *manuscript*, December 1999.

[10] B. Chazelle, "Triangulating a Simple Polygon in Linear Time", *Discrete Comput. Geometry*, 6 (1991), pp. 485-524.

[11] D. Chetverikov and A. Lerch, "A Multiresolution Algorithm for Rotation-Invariant Matching of Planar Shapes", *Pattern Rec. Letts.* 13 (1992), pp. 669-676.

[12] K. Chong and S. Sahni, "Optimal Realizations of Floorplans", *IEEE Trans. CAD* 12(6) (1993), pp. 793-801.

[13] W. Dai and E. S. Kuh, "Simultaneous Floor Planning and Global Routing for Hierarchical Building Block Layout", *IEEE Trans. CAD* 6(5) (1987), pp. 828-837.

[14] H. Esbensen and E. S. Kuh, "Design Space Exploration Using the Genetic Algorithm", *Proc. ISCAS*, 1996, vol. 4, pp. 500-503.

[15] T. Gonzalez and M. Razzazi, "On the Generalized Channel Definition Problem", *Proc. Great Lakes Symp. on VLSI*, 1991, pp.88-91.

[16] T. Gonzalez and S. Zheng, "Approximation Algorithms for Partitioning a Rectangle with Interior Points", *Algorithmica* 5 (1990), pp. 11-42.

[17] J. Grason, *Methods for the Computer-Implemented Solution of a Class of "Floor Plan" Design Problems*, Ph.D. Thesis, CMU Electrical Engineering Dept., May 1970.

[18] P.-N. Guo, C.-K. Cheng and T. Yoshimura, "An O-Tree Representation of Non-Slicing Floorplan and its Applications", *Proc. DAC*, 1999, pp. 268-273.

[19] K. Hayashi, M. Inoue, T. Masuzawa and H. Fujiwara, "A Layout Adjustment Problem for Disjoint Rectangles Preserving Orthogonal Order", *Proc. 6th Intl. Symp. on Graph Drawing*, 1998, pp. 183-197.

[20] H.J.A.M. Heijmans and A. Tuzikov, "Similarity and Symmetry Measures for Convex Shapes Using Minkowski Addition", *IEEE Trans. on PAMI* 20 (1998), pp. 980-993.

[21] T. C. Hu, "Physical Design: Mathematical Models and Methods", keynote address, *Proc. ISPD*, 1997, pp. 207-210.

[22] D.P. Huttenlocher, G.A. Klanderman and W.J. Rucklidge, "Comparing Images Using the Hausdorff Distance", *IEEE Trans. on PAMI* 15 (1993), pp. 850-863.

[23] T. Kong and X.L. Hong, "Timing-Driven Floorplanning Algorithm for Building Block Layout", (*Fourth Intl. Conference on Computer-Aided Design and Computer Graphics*, 1995) *Proc. SPIE* vol. 2644, 1996, pp. 477-482.

[24] E. Koutsoupias, C. H. Papadimitriou and M. Sideri, "On the Optimal Bisection of a Polygon", *ORSA J. Computing* 4(4) (1992), pp. 435-438.

[25] H. Krupnova, A. Abbara and G. Saucier, " A Hierarchy-Driven FPGA Partitioning Method", *Proc. DAC*, 1997, pp. 522-525.

[26] F.P. Kuhl and C.P. Giardina, "Elliptic Fourier Features of a Closed Contour", *Computer Graphics and Image Processing* 18 (1982), pp. 236-258.

[27] M.-T. Kuo and C.-K. Cheng, "A Network Flow Approach for Hierarchical Tree Partitioning", *Proc. DAC*, 1997, pp. 512-517.

[28] T.-C. Lee, "A Bounded 2D Contour Searching Algorithm for Floorplan Design With Arbitrarily Shaped Rectilinear and Soft Modules", *Proc. DAC*, 1993, pp. 525-530.

[29] T. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout*, New York, Wiley-Teubner, 1990.

[30] T. Lengauer and R. Muller, "Robust and Accurate Hierarchical Floorplanning with Integrated Global Wiring", *IEEE Trans. on CAD* 12(6) (1993), pp. 802-809.

[31] C. Levcopoulos, "Fast Heuristics for Minimum Length Rectangular Partitions of Polygons" *Proc. ACM Symp. on Computational Geometry*, 1986, pp. 100-108.

[32] C. Levcopoulos and A. Ostlin, "Linear-Time Heuristics for Minimum Weight Rectangulation", *Proc. 5th Scandinavian Workshop on Algorithm Theory*, 1996, pp. 271-283.

[33] C. C. Lin and R. Chellappa, "Classification of Partial 2-D Shapes Using Fourier Descriptors", *IEEE Trans. on PAMI* 9 (1987), pp. 686-690.

[34] W. T. Liou, J. J. Tan and R. C. T. Lee, "Minimum Rectangular Partition Problem for Simple Rectilinear Polygons", *IEEE Trans. on CAD* 9(7) (1990), pp. 720-733.

[35] T.-L. Liu and D. Geiger, "Approximate Tree Matching and Shape Similarity", *Proc. IEEE Intl. Conference on Computer Vision*, 1999, vol. 1, pp. 456-462.

[36] F. Mokhtarian and A. Mackworth, "Scale-Based Description and Recognition of Planar Curves and Two-Dimensional Shapes", *IEEE Trans. on PAMI* 8 (1986), pp. 34-43.

[37] H. Murata, K. Fujiyoshi, S. Nakatake and Y. Kajitani, "Rectangle-Packing-Based Module Placement", *Proc. ICCAD*, 1995, pp. 472-479.

[38] S. Nakatake, K. Fujiyoshi, H. Murata and Y. Kajitani, "Module Placement on BSG-Structure and IC Layout Applications", *Proc. ICCAD*, 1996, pp. 484-91.

[39] M. Ohmura, S. Wakabayashi, Y. Toyohara, J. Miyao and N. Yoshida, "Hierarchical Floorplanning and Detailed Global Routing with Routing-Based Partitioning", *Proc. ISCAS*, 1990, pp. 1640-1643.

[40] K. Okada, T. Yamanouchi and T. Kambe, "Rectilinear Shape Formation Method on Block Placement", *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences* E81-A(3) (1998), pp. 446-454.

[41] H. Onodera, Y. Taniguchi and K. Tamaru, "Branch-and-Bound Placement for Building Block Layout", *Proc. DAC*, 1991, pp. 433-439.

[42] R. H. J. M. Otten, "Automatic Floorplan Design", *Proc. DAC*, 1982, pp. 261-267.

[43] R. H. J. M. Otten, "Graphs in Floorplan Design", *Int. J. Circuit Theory Applic.* 16 (1988), pp. 391-410.

[44] F. P. Preparata, M. I. Shamos, *Computational Geometry: An Introduction*, New York, Springer-Verlag, 1985.

[45] A. Ranjan, K. Bazargan and M. Sarrafzadeh, "Floorplanner 1000 Times Faster: A Good Predictor and Constructor", *Proc. Intl. Workshop on System-Level Interconnection Prediction (SLIP)*, 1999, pp. 115-120.

[46] L. Sha and R. W. Dutton, "An Analytical Algorithm for Placement of Arbitrarily Sized Rectangular Blocks", *Proc. DAC*, 1985, pp. 602-608.

[47] S. J. Souri and K. C. Saraswat, "Interconnect Performance Modeling For 3D Integrated Circuits with Multiple Si Layers", *Proc. Intl. Interconnect Technology Conference*, 1999, pp. 24-26.

[48] L. Stockmeyer, "Optimal Orientations of Cells in Slicing Floorplan Designs", *Info. and Control* 59 (1983), pp. 91-101.

[49] H. Su, A. C. Wu and Y. Lin, "Performance-Driven Soft-Macro Clustering and Placement by Preserving HDL Design Hierarchy", *Proc. ISPD*, 1998, pp. 12-17.

[50] H. Su, A. C. Wu and Y. Lin, "A Timing-Driven Soft-Macro Resynthesis Method in Interaction with Chip Floorplanning", *Proc. DAC*, 1999, pp. 262-267.

[51] Y. Tsay, W. Fang, A. C. Wu and Y. Lin, "Preserving HDL Synthesis Hierarchy for Cell Placement", *Proc. ISPD*, 1997, pp. 169-174.

[52] T. C. Wang and D. F. Wong, "Optimal Floorplan Area Optimization", *IEEE Trans. on CAD* 11(8) (1992), pp. 992-1002.

[53] S. Wimer, I. Koren and I. Cederbaum, "Optimal Aspect Ratios of Building Blocks in VLSI", *IEEE Trans. on CAD* 8(2) (1989), pp. 139-145.

[54] T. Yamanouchi, K. Tamakashi and T. Kambe, "Hybrid Floorplanning Based on Partial Clustering and Module Restructuring", *Proc. ICCAD*, 1996, pp. 478-483.

[55] H.H. Yang and D.F. Wong, "Efficient Network Flow Based Min-Cut Balanced Partitioning", *IEEE Trans. on CAD* 15 (1996), pp. 1533-1540.

[56] F.Y. Young and D.F. Wong, "How Good are Slicing Floorplans?", *Integration* 23(1) (1997), pp. 61-73.

[57] K. H. Yeap and M. Sarrafzadeh, "An Integrated Algorithm for Optimal Floorplan Sizing and Enumeration", *Proc. European Design Automation Conf.*, 1993, pp. 29-33.