Futures for Partitioning in Physical Design[*]

Andrew B. Kahng

UCLA Computer Science Department, Los Angeles, CA 90095-1596
abk@cs.ucla.edu, http://vlsicad.cs.ucla.edu

Abstract

The context for partitioning in physical design is dominated by two concerns: top-down design and the focus on spatial embedding. The role of partitioning is exactly that of a facilitator of divide-and-conquer metaheuristics for floorplanning, timing and placement optimization. Formulations or optimization objectives for partitioning follow from its context and role. Finally, the available algorithm technology determines how effectively we can address a given partitioning formulation and optimize a given objective. This invited paper considers the future of partitioning for physical design in light of these factors, and proposes a list of technology needs. A living version of this paper can be found at vlsicad.cs.ucla.edu .

1 Introduction

Partitioning in VLSI physical design has been actively studied over the past two decades, with hundreds of works in the literature. A comprehensive review can be obtained from [3] [13] [19] [8]. The standard formulation seeks a multi-way partitioning of a vertex- and hyperedge-weighted directed hypergraph, with a minimum hyperedge-cut objective subject to partition size constraints. Recent years have seen several works that add I/O-count or path-cut constraints, replication and retiming degrees of freedom, hierarchy awareness, etc. to the standard formulation. However, the bulk of the literature still focuses on "min-cut" bipartitioning.

As process technology advances, the *context* for partitioning in physical design will change. Huge device complexities and short product cycles force *top-down* hierarchical approaches to design synthesis, validation and reuse. At the same time, more design activity is *"physical"*, i.e., dependent on knowing the eventual spatial embedding of devices and interconnects. Because physical phenomena (delay, noise, power, reliability, manufacturability) become more difficult to abstract, we see other changes to design methodology, e.g., tighter links between analysis and synthesis, and fewer demarcations between "design phases". The net result is a stronger emphasis on *prediction* and *convergence*.

Within a given design methodology context, the *role* of partitioning should also be understood. Partitioning is simply a facilitator of the divide-and-conquer approach. Its purpose is to decompose problems in a manner appropriate to the application, *without losing solution quality*. Increasingly, a large part of "solution quality" depends on preserving the previous activity that led up to the current state of the design. Hence, a critical future requirement for a partitioner is that it completely captures, and never loses, "design state" information.[1] Many challenges, such as handling constraints and incorporating "non-local" objective function terms, lie ahead as partitioners are matched more closely to their applications.

Understanding the future top-down physical design context and the role of partitioning allows us to identify future *problem formulations and objectives*. From these, we can identify the *algorithm technology* that is capable of delivering high-quality solutions.[2] The remainder of this paper gives some perspectives on these elements, along with a list of technology needs.

2 The Top-Down Design Context

Top-down design relies on *predictions* of the achievable envelopes of solutions. Without predictions, we are reduced to "constructive estimates" (e.g., run the placer to predict what the placement looks like), which are typically too slow for effective design optimization.[3] Predictive models must also be accurate, else too much solution quality is left on the table.[4] Finally, predictive models must be usable as terms in a design optimization objective. Today, design synthesis tools are barely able to incorporate bottom-up analysis macromodels (e.g., analytic crosstalk noise approximations) within design synthesis objectives. The future will require *instance models* and *tool models* that allow prediction of a given tool's output according to instance parameters, sensitivity to control parameters, CPU resource, etc. Such models must be embeddable within optimization objectives.

Also implicit in top-down design is the notion of *convergence*, which is achieved by forward-annotating both constraints[5] and knowledge about the design state. Some constraint types translate well into traditional partitioning formulations (e.g., a pre-synthesis floorplanner will forward-annotate region constraints into a flat partitioning-based placer), while others do not (e.g., budgeted path timing constraints, or alignment constraints in placement). Examples of knowledge about the design state include: structure of clock and test implementation that is temporarily removed from a place-and-route input; knowledge of datapath functions within a synthesized random-logic netlist, the floorplanning tool's assumptions regarding layer assignment of chip-level routes, simulation traces that provide switching activity profiles, etc. Today's partitioning tools cannot easily capture such knowledge, even when co-

[*]Research supported by a grant from Cadence Design Systems, Inc.

[1]For example, knowledge of scan groups or clock phases may be impossible to capture in the partitioning input semantics. Another example is the fact that a cluster represents a 16-bit datapath function and should be split only along the data- or control-flow axes; this, along with the associated alignment constraint in placement, is extremely difficult to capture for today's min-cut partitioners.

[2][3] and [13] also provide lists of future directions for partitioning.

[3]An exception is when the algorithm runtime scales near-linearly with instance complexity. In this case, running the algorithm is a bona fide estimation strategy; cf. multilevel partitioners and multigrid sparse system solvers for placement.

[4]The challenge of *accurately* abstracting deep-submicron physical phenomena has led to an interesting bifurcation in the architecture of RTL-down chip implementation tools. Broadly speaking, one approach is to use *unifications*, e.g., unifying spatial and temporal optimizations, or unifying analysis, (re-)synthesis and (re-)specification activities either within a given design phase or across several design phases. A typical consequence is the "unified design database" and the "(timing) analysis backplane" architecture that simultaneously supports logic synthesis, timing optimization, and place-and-route with an incremental "construct by correction" use model. The other approach is to continue to develop new *abstractions* that decouple, e.g., logic synthesis from layout synthesis, place-and-route from timing optimization, etc. Typical mechanisms are often equated with *methodology*, e.g., the "predictor-adapter" or "constant-delay" approach in logic synthesis, or the use of slew time control to handle noise problems in placement-based circuit optimization. Typical consequences are more linear, "correct by construction" design flows.

[5]By this, I mean the constraints that can arise from budgeting operations, from previous constructions (e.g., one might wish to preserve as much as possible the floorplan or placement achieved in a previous design iteration), from predictions, and even from *assumptions*. The latter two types of constraints arise, in, e.g., predictor-adapter approaches to predictive modeling.

erced via weighting, clustering and constraints. Yet, future applications will be dominated by such forward propagation of constraints and knowledge.

3 The Spatial Embedding Context

Improved manufacturing technology expands the scope of physical design, since the effects of *spatial embedding* on performance and function must be understood earlier in the design process. Fundamentally, spatial embedding is *placement*; the many flavors are a result of different constraints, objective functions, and instance structure in given applications. A few examples:

- Placement for MCMs and prototyping architectures has unique difficulties due to the highly non-uniform nature of the device and interconnect resources. One often encounters *hard constraints* (e.g., fixed I/O and device capacities), *non-geometric* objective function terms (e.g., from architecture-dependent costs of different routing resources), and *non-local* objective function terms (e.g., minimize the maximum number of hops between FPGA devices on any latch-to-latch path). Similar difficulties arise in what might be termed *floorplan-driven partitioning*, for which antecedents were studied in [27].

- Placement for pre-synthesis floorplanning presents a mix of hard, (estimated) semi-hard and (estimated) soft blocks. Aspects of the input (components, connectivity, time budgets, etc.) are only *partially* or *probabilistically* specified (see recent work of Sarrafzadeh and coauthors, e.g., [5]).[6] Block floorplanning also exemplifies the "phase transition" between heterogeneous and homogeneous instances, and discrete and continuous perspectives for optimization. Finally, in most methodologies the reconciliation of functional and physical hierarchies during block placement (as well as chip-level route planning, pin definition and performance optimization) must be maintained. Certain node clusterings or smashings might be prohibited. This leads to new partitioning formulations (see the recent work of Cheng and coauthors, e.g., [17]).

- Gate-level placement in row-based implementation styles presents nearly geometric objective function terms, along with very homogeneous instance structure. However, as noted above, even these placement instances will become constraint-dominated (e.g., the floorplanner will forward-annotate estimated locations, route topologies, timing and noise budgets, etc.).

4 The Role of Partitioning

Partitioning facilitates the divide-and-conquer approach by decomposing problems in a manner appropriate to the application without losing too much solution quality. This is a "mission statement" for the partitioning field. Today, partitioning is a key element of many heuristics for physical design applications. However, it is not clear whether today's "standard" partitioning algorithms (basically, multilevel implementations of Fiduccia-Mattheyses (FM) variants [10] [4] [16]) will successfully adapt to tomorrow's challenges: non-local and non-geometric objectives; constraint-dominated formulations; the need to maintain "design state" information (noted above); etc. I believe that the partitioning field must (i) carefully determine the limits of the major partitioning approaches, and (ii) never lose sight of the "mission statement".

A basic precept that follows from the mission statement is that partitioning objectives and algorithms must be fitted to applications, not the other way around. To illustrate the mismatches that can occur between partitioning algorithms and the underlying application domain, consider the following synthesis of row-based placer evolution.

[6] Lillis [20] observes that if block attributes such as area are represented by probability distributions, a feasible partitioning could be defined as one where each partition contains nodes whose area satisfies capacity constraints with some pre-specified "high probability".

On one hand, top-down partitioning-based placers have become popular for their speed advantage. This speed comes from using partitions that are a *coarse-grain abstraction* of the layout region and its spatial, geometric properties. For example, the first bisection of a min-cut placer is placing the entire (flat, fine-grain) netlist onto just two points(!) This extreme coarseness in the *layout model* – vis-a-vis the minimum-wirelength placement objective – is the mismatch that has motivated terminal propagation [9], quadrisection [25], quadrisection with exact placement objective [12], etc. The natural extension of this trend – toward top-down 8-way, 16-way, etc. partitioning with "exact placement objective" – fails because the number of gain buckets in FM variants grows quadratically with k in k-way partitioning, and the size of the "net gain vector" [12] grows exponentially in k. Hence, the correct extension of the trend is to *non-partitioning based* placers with a less coarse-grain layout abstraction.[7]

On the other hand, flat (annealing-based) placers traditionally have fine-grain views of both the netlist and the layout. They obtain speedups via (multilevel) clustering [26]; with this technique, the flat placer can be viewed as mapping (1) a coarse-grain (clustered) netlist onto (2) a fine-grain layout target. Notice the contrast with the trend for top-down partitioning-based placers. The critical observation is that the two classical placer speedup techniques – coarsening the *netlist*, and coarsening the *layout target* – are orthogonal. With this observation, one realizes the import of the recent NRG placement approach of Sarrafzadeh and Wang [22]: it shows that the two types of speedup can and should be *independently* implemented. Using these ideas to achieve optimum speedup and solution quality is a research direction that will, among other things, determine the limits of FM-style methods for multi-way partitioning.

5 Future Research Directions

This paper concludes by listing directions for future research. Readers may find it interesting to compare what follows with the predictions given three years ago in [3] and two years ago in [13].[8]

- Prediction in top-down design requires *instance models*. We need structural models of *heterogeneous* design instances, at all levels of representation, that go beyond "single-number" characterizations. Note that traditional measures such as I/O count, gate count, latch count, diameter and Rent parameter all use single numbers to describe the design. Today's Rent-like characterizations are based solely on *topological* structure; future models must capture *delay* and *temporal* structure[9] as well as *communication* and *function* complexity. Such new structural models will also facilitate such design activities as *interconnect tuning* [14].

- Prediction requires *tool models* as well. We need models of global optimization metaheuristics that allow prediction of output solution quality based on instance parameters and available CPU resource. Simple examples include (i) "best-so-far" curves for such iterative global optimizations as simulated annealing (based on models of optimization cost surfaces and neighborhood structures), and (ii) distributions of solution quality for multi-start metaheuristics (e.g., order statistics of multiple runs of FM partitioning).

[7] The min-cut framework suffers from other mismatches with the placement application, e.g., "min-cuts" typically create underutilized routing resources along cutlines.

[8] Both surveys [3] and [13] point out the need for improved benchmarks and "culture" (reporting of experimental protocols and results; sharing and reuse of ideas/software across multiple application domains). Alpert [1] has made a huge step toward improved benchmarks, and though cultural gaps still persist, I will not address them below. Both surveys point to constraint-dominated partitioning, and the potential for combinatorial or enumerative methods. [13] suggests logic resynthesis and estimation of system properties; [3] suggests the study of clustering and iterative metaheuristics for global optimization. All of these futures are mentioned again below, hopefully with some fresh perspectives. The one direction that seems to have been dropped is spectral partitioning, which has seen significant advances by researchers at, e.g., UC Santa Cruz and the University of Waterloo, but which may not be critical in the future [2].

[9] For example, we may characterize edges according to their feasible or budgeted delays.

- To support the previous goal, various partitioning formulations must be characterized as large-scale discrete global optimizations. What are the statistics of the optimization cost surface under given neighborhood structures? What is the appropriate choice of metaheuristics (e.g., multi-start greed, annealing, large-step Markov chains, population-based search, etc.) for search in a given cost surface under CPU bounds? Studies of iterative global optimization [6] suggest that iterated-descent methods and "go-with-winners" types of population-based search are very effective. The theory of problem-space and heuristic-space methods [24][10] is also of particular interest, given the possibility of subtle mismatches between the partitioning objective and the application's true objective. A new theory must also be developed that allows us to effectively introduce *corrections* for such mismatches.[11]

- Formulations for partitioning that explicitly account for spatial embedding must be developed. As noted above, floorplan-driven partitioning and embedding into rapid prototyping architectures provide examples of highly non-geometric objective function terms. Combinatorial techniques seem more appropriate that iterative methods for such formulations. The links between placement and partitioning can also become more bidirectional, i.e., placement can be a tool that leads to good partitioning solutions, just as partitioning is a tool for placement.

- Formulations that handle *timing* constraints must be developed. Lillis [20] notes that recent retiming objectives may well give way to, e.g., minimization of maximum latch-to-latch hops across partition boundaries. Above, this was cited as one example of a "non-local" objective function term that is not naturally handled by iterative FM-style partitioners. Algorithms – likely combinatorial in nature – that handle such non-local objectives must be identified.

- The combinatorial structure of partitioning must be understood. When partitioning with tight balance constraints and irregular module sizes [1], the continuous min-cut formulation gives way to a more discrete packing formulation, and the precise nature of this "phase transition" needs to be clarified. For example, understanding the sensitivity of partitioners to size imbalances, compatibility of module sizes within partition size bounds, and uncertainty of the design will have implications for how we cluster during RT-level design planning.

- The limits of *multilevel* variants of the FM heuristic must be determined. Given the synthesis of placer evolution above, there is a clear research question: At what point will metaheuristics with different neighborhood structure (e.g., multilevel clustered annealing) achieve better k-way partitioning and/or placement solutions than k-way or hierarchical top-down multilevel FM-based partitioning?[12] More generally, combinatorial algorithms (notably network flows for replication cuts, balanced min-cuts, and linear placement), mathematical programming, and continuous optimization techniques must be rapidly explored as potential replacements for iterative approaches.

- Understanding of clustering must improve. With respect to physical design, there are at least three major purposes for clustering. First, clustering contracts a large problem instance into a smaller instance to save runtime or allow better search of the solution space; this is the primary motivation in the literature. Second, when it is known that modules should be grouped together, clustering them together prevents, e.g., a top-down partitioning placer from making a mistake. Third, and most important, clustering captures design knowledge that would otherwise be forgotten (e.g., identification of synthesized buffer-inverter trees, two-dimensional datapath regularity [21], clock/test structures, electrical clustering,[13] or hierarchy). To date, principled studies of clustering have not yet been published (some first steps are given in [15] [11] [16]). At least three threads of research remain disconnected: clustering objectives (which have mostly been ad hoc), clustering heuristics (none of which yet directly optimizes any of the proposed clustering objectives), and benefits to given applications (which currently lack metrics as well as direct links back to the clustering objectives). These should be unified.

  Other research topics: (1) How much does the choice of clustering heuristic matter in multilevel partitioning and placement approaches? (2) If the choice does matter, what clustering heuristic and parameters (e.g., the matching ratio in heavy-edge matching [16]) should be used for circuits with given characteristics, and given partitioning/placement applications? (3) What are the best approaches to timing-oriented clustering? (Lillis [20] suggests concepts of convexity (no directed path leaves and later re-enters a cluster) and quasi-convexity.)

[10] Lillis [20] has explored FM partitioning in a secondary *general connectivity* graph. The idea is that good solutions in the secondary graph can provide good starting points for FM in the original graph; this recalls motivations for two-phase and multilevel FM.

[11] Recall the mismatch between top-down min-cut partitioning and the min-wirelength objective in row-based placement. If we can bipartition according to the min-cut objective, and anneal according to the "true" (min-wirelength) placement objective, what is the best strategy for interpolating annealing "corrections" into a top-down bipartitioning placement? Some first steps toward resolving this issue have been taken in [7].

[12] Yeh et al. [28] observe that annealing is inferior to FM for 2-way partitioning. Our experience [7] suggests that the "crossover point" occurs at surprisingly low values of k.

6 Acknowledgments

Over the years, I have enjoyed stimulating discussions of partitioning and clustering with: Lars Hagen, Chuck Alpert, Ken Boese, Dennis Huang, Rahul Sharma, Andy Caldwell and Igor Markov at UCLA; Chung-Kuan Cheng, John Lillis and Arthur Wei at UCSD; Ren-Song Tsay at ArcSys; George Janac at HLDS/Cadence; Wenyong Deng, David Yao, Limin He, Lou Scheffer, Antun Domic, Phil Tu and Shantanu Dutt at Cadence; and Jon Frankle at Xilinx/Aristo. During the (too-short) preparation of this paper, John Lillis and Chuck Alpert contributed their thoughts on partitioning futures, and valuable feedback on the draft. Without these influences and the invitation by Majid Sarrafzadeh to organize the ISPD-98 tutorial panel, this paper would not exist.

REFERENCES

[1] C. J. Alpert, *personal communication*, February 1998.

[2] C. J. Alpert, "The ISPD Circuit Benchmark Suite", *Proc. ACM/IEEE Intl. Symp. on Physical Design*, April 1998.

[3] C. J. Alpert and A. B. Kahng, "Recent Directions in Netlist Partitioning: A Survey", *Integration* 19 (1995), pp. 1-81.

[4] C. J. Alpert, J.-H. Huang and A. B. Kahng, "Multilevel Circuit Partitioning", *Proc. ACM/IEEE Design Automation Conference*, 1997, pp. 530-533.

[5] K. Bazargan, S. Kim and M. Sarrafzadeh, "Nostradamus: A Floorplanner of Uncertain Design", *Proc. ACM/IEEE Intl. Symp. on Physical Design*, April 1998.

[6] K. D. Boese, *Models for Iterative Global Optimization*, Ph.D. Thesis, UCLA Computer Science Dept., 1996.

[7] A. E. Caldwell, A. B. Kahng and I. L. Markov, *manuscript*, 1997.

[8] W. E. Donath, "Logic Partitioning", in B. Preas and M. Lorenzetti, eds., *Physical Design Automation in VLSI Systems*, Benjamin/Cummings, 1988.

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

[13] Antun Domic points out the example of a weak driver output connected to the input of a repeater, whose output goes to a receiver. A partitioner sees two 2-pin nets that have equal cut cost. However, it is better to group the weak driver with the repeater.

[10] C. M. Fiduccia and R. M. Mattheyses, "A Linear Time Heuristic for Improving Network Partitions", *Proc. ACM/IEEE Design Automation Conference*, 1982, pp. 175-181.

[11] S. Hauck and G. Borriello, "An Evaluation of Bipartitioning Techniques", *IEEE Trans. on Computer-Aided Design* 16 (1997), pp. 849-866.

[12] D. J. Huang and A. B. Kahng, "Partitioning-Based Standard-Cell Global Placement with an Exact Objective", in *Proc. ACM/IEEE Intl. Symp. on Physical Design*, Napa, April 1997, pp. 18-25.

[13] F. R. Johannes, "Tutorial: Partitioning of VLSI Circuits and Systems", *Proc. ACM/IEEE Design Automation Conference*, 1996, pp. 83-87.

[14] A. B. Kahng, S. Muddu, E. Sarto and R. Sharma, "Interconnect Tuning Strategies for High-Performance ICs", *Proc. Design, Automation and Test in Europe (DATE)*, Paris, February 1998.

[15] A. B. Kahng and R. Sharma, "Studies of Clustering Objectives and Heuristics for Improved Standard-Cell Placement", *manuscript*, 1997.

[16] G. Karypis, R. Aggarwal, V. Kumar and S. Shekhar, "Multilevel Hypergraph Partitioning: Applications in VLSI Domain", *Proc. ACM/IEEE Design Automation Conference*, 1997, pp. 526-529.

[17] M.-T. Kuo, L.-T. Liu and C.-K. Cheng, "Network Partitioning Into Tree Hierarchies", *Proc. ACM/IEEE Design Automation Conference*, 1996, pp. 477-482.

[18] E. L. Lawler, K. N. Levitt and J. Turner, "Module Clustering to Minimize Delay in Digital Networks", *IEEE Trans. on Computers* 18 (1969), pp. 47-57.

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

[20] J. Lillis, *personal communication*, February 1998.

[21] R. X. T. Nijssen and J. A. G. Jess, "Two-Dimensional Datapath Regularity Extraction", *Proc. ACM/SIGDA Physical Design Workshop*, April 1996, pp. 111-117.

[22] M. Sarrafzadeh and M. Wang, "NRG: Global and Detailed Placement", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1997, pp. 532-537.

[23] Semiconductor Industry Association, "The National Technology Roadmap for Semiconductors: Technology Needs", December 1997.

[24] R. H. Storer, S. D. Wu and R. Vaccari, "New Search Spaces for Sequencing Problems With Application to Job Shop Scheduling", *Management Science* 38 (1992), pp. 1495-1509.

[25] P. R. Suaris and G. Kedem, "Quadrisection: A New Approach to Standard Cell Layout," *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 1987, pp. 474-477.

[26] W. Sun and C. Sechen, "Efficient and Effective Placements for Very Large Circuits", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 1993, pp. 170-177.

[27] G. Vijayan, "Generalization of Min-Cut Partitioning to Tree Structures and Its Applications", *IEEE Trans. on Computers* 40(3) (1991), pp. 307-314.

[28] C.-W. Yeh, C.-K. Cheng and T.-T. Y. Lin, "Optimization by Iterative Improvement: An Experimental Evaluation on Two-Way Partitioning", *IEEE Transactions on Computer-Aided Design* 14 (1995), pp. 145-153.