Machine Learning Applications in Physical Design: Recent Results and Directions

Andrew B. Kahng
CSE and ECE Departments, UC San Diego, La Jolla, CA 92093
abk@ucsd.edu

ABSTRACT

In the late-CMOS era, semiconductor and electronics companies face severe product schedule and other competitive pressures. In this context, electronic design automation (EDA) must deliver "design-based equivalent scaling" to help continue essential industry trajectories. A powerful lever for this will be the use of machine learning techniques, both inside and "around" design tools and flows. This paper reviews opportunities for machine learning with a focus on IC physical implementation. Example applications include (1) removing unnecessary design and modeling margins through correlation mechanisms, (2) achieving faster design convergence through predictors of downstream flow outcomes that comprehend both tools and design instances, and (3) corollaries such as optimizing the usage of design resources licenses and available schedule. The paper concludes with open challenges for machine learning in IC physical design.

ACM Reference Format:
Andrew B. Kahng. 2018. Machine Learning Applications in Physical Design: Recent Results and Directions. In *ISPD'18: 2018 International Symposium on Physical Design, March 25–28, 2018, Monterey, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3177540.3177554

1 CONTEXT: THE LAST SCALING LEVERS

Semiconductor technology scaling is challenged on many fronts that include pitch scaling, patterning flexibility, wafer processing cost, interconnect resistance, and variability. The difficulty of continuing Moore's-Law lateral scaling beyond the foundry 5nm node has been widely lamented. Scaling boosters (buried interconnects, backside power delivery, supervias), next device architectures (VGAA FETs), ever-improving design-technology co-optimizations, and use of the vertical dimension (heterogeneous multi-die integration, monolithic 3D VLSI) all offer potential extensions of the industry's scaling trajectory. In addition, various "rebooting computing" paradigms – quantum, approximate, stochastic, adiabatic, neuromorphic, etc. – are being actively explored.

No matter how future extensions of semiconductor scaling materialize, the industry already faces a crisis: design of new products in advanced nodes costs too much.[1] Cost pressures rise when incremental technology and product benefits fall. Transitioning from 40nm to 28nm brought as little as 20% power, performance or area (PPA) benefit. Today, going from foundry 10nm to 7nm, or from 7nm to 5nm, the benefit is significantly less, and products may well realize only one – possibly two – of these PPA wins. The 2013 ITRS roadmap [40] highlighted a gap between scaling of *available* transistor density and scaling of *realizable* transistor density. This *design capability gap*, which adds to the spotlight on design cost, is illustrated in Figure 1 [20]. The recent DARPA Intelligent Design of Electronic Assets (IDEA) [38] program directly calls out today's design cost crisis, and seeks a "no human in the loop," 24-hour design framework for RTL-to-GDSII layout implementation.

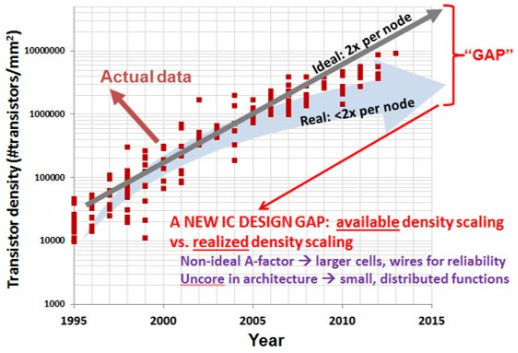

Figure 1: Design Capability Gap [40] [20].

More broadly, the industry faces three intertwined challenges: cost, quality and predictability. *Cost* corresponds to engineering effort, compute effort, and schedule. *Quality* corresponds to traditional power, performance and area (PPA) competitive metrics along with other criteria such as reliability and yield (which also determines cost). *Predictability* corresponds to the reliability of the design schedule, e.g., whether there will be unforeseen floorplan ECO iterations, whether detailed routing or timing closure flow stages will have larger than anticipated turnaround time, etc. Product quality of results (QOR) must also be predictable. Each of three challenges implies a corresponding "last lever" for scaling. In other words, reduction of design cost, improvement of design quality, and reduction of design schedule (which is the flip side of predictability; recall that Moore's Law is "one week equals one percent") are are all forms of *design-based equivalent scaling* [19] [20] that can extend availability of leading-edge technology to designers and new products. A powerful lever for this will be the use of machine learning (ML) techniques, both inside and "around" electronic design automation (EDA) tools.

The remainder of this paper reviews opportunities for machine learning in IC physical implementation. Section 2 reviews example ML applications aimed at removing unnecessary design and modeling margins through new correlation mechanisms. Section 3 reviews applications that seek faster design convergence through predictors of downstream flow outcomes. Section 4 gives a broader vision of how ML can help the IC design and EDA fields escape the current "local minimum" of coevolution in design methodology and design tools. Section 5 concludes with open challenges for ML in IC physical design. Since this paper shares its subject matter and was written contemporaneously with [23], readers are referred to [23] for additional context.

[1] The 2001 *International Technology Roadmap for Semiconductors* [40] noted that "cost of design is the greatest threat to continuation of the semiconductor roadmap".

ISPD'18, March 25–28, 2018, Monterey, CA, USA

ACM ISBN 978-1-4503-5626-8/18/03…$15.00
https://doi.org/10.1145/3177540.3177554

2 IMPROVING ANALYSIS CORRELATION

Analysis miscorrelation exists when two different tools return different results for the same analysis task (parasitic extraction, static timing analysis (STA), etc.) even as they apply the same "laws of physics" to the same input data. As illustrated in Figure 2, better accuracy always comes at the cost of more computation. [2] Thus, miscorrelation between two analysis reports is often the inevitable consequence of runtime efficiency requirements. For example, signoff timing is too expensive (tool licenses, incremental analysis speed, loops of timing window convergence, query speed, number of corners, etc.) to be used within tight optimization loops.

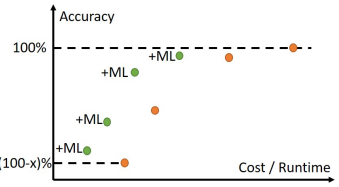

Figure 2: Accuracy-cost tradeoff in analysis.

Miscorrelation forces introduction of design guardbands and/or pessimism into the flow. For example, if the place-and-route (P&R) tool's STA report determines that an endpoint has positive worst setup slack, while the signoff STA tool determines that the same endpoint has negative worst slack, an iteration (ECO fixing step) will be required. On the other hand, if the P&R tool applies pessimism to guardband its miscorrelation to the signoff tool, this will cause unneeded sizing, shielding or VT-swapping operations that cost area, power and design schedule. Miscorrelation of timing analyses is particularly harmful: (i) timing closure can consume up to 60% of design time [12], and (ii) added guardbands not only worsen power-speed-area tradeoffs [3, 9, 12], but can also lead to non-convergence of the design.

Signoff Timer Correlation. Correlation to signoff timing is the most valuable target for ML in back-end design. Improved correlation can give "better accuracy for free" that shifts the cost-accuracy tradeoff (i.e. achieving the ML impact in Figure 2) and reduces iterations, turnaround time, overdesign, and tool license usage along the entire path to final design signoff.[3] [27] uses a learning-based approach to fit analytical models of wire slew and delay to estimates from a signoff STA tool. These models improve accuracy of delay and slew estimations along with overall timer correlation, such that fewer invocations of signoff STA are needed during incremental gate sizing optimization [34]. [16] applies deep learning to model and correct divergence between different STA tools with respect to flip-flop setup time, cell arc delay, wire delay, stage delay, and path slack at timing endpoints. The approach achieves substantial (multiple stage delays) reductions in miscorrelation. Both a one-time training methodology using artificial and real circuit topologies, as well as an incremental training flow during production usage, are described (Figure 3(a)). [30] achieves accurate (sub-10ps worst-case error in a foundry 28nm FDSOI technology) prediction of SI-mode timing slacks based on "cheaper, faster" non-SI mode reports. A combination of electrical, functional and topological parameters are used to predict the incremental transition times and arc/path delays due to SI effects. From this and other works, an apparent "no-brainer" is to use Hybrid Surrogate Modeling (HSM) [28] to combine predicted values from multiple ML models into final predictions (Figure 3(b)).

[2] The figure's y-axis shows that the error of the simplest estimates (e.g., "Elmore delay") can be viewed as having accuracy of $(100 - x)\%$. The return on investment for new ML applications would be higher when x is larger.

[3] Given that miscorrelation equates with margin, it is useful to note [18].

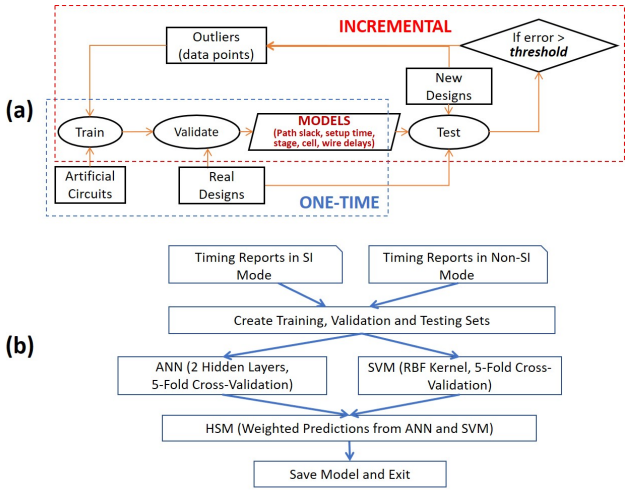

Figure 3: Flow and results for machine learning of STA tool miscorrelation: (a) [16]; (b) [30]. HSM approaches are described in [28] [29].

Next Targets. [23] identifies two near-term extensions in the realm of timer analysis correlation. **(1) PBA from GBA.** Timing analysis pessimism is reduced with *path-based analysis* (PBA), at the cost of significantly greater runtime than traditional *graph-based analysis* (GBA). In GBA, worst (resp. best) transitions (for max (resp. min) delay analyses) are propagated at each pin along a timing path, leading to conservative arrival time estimates. PBA calculates path-specific transition and arrival times at each pin, reducing pessimism that can easily exceed a stage delay. Figure 4 shows the frequency distribution of endpoint slack pessimism in GBA. This pessimism harms the design flow, e.g., when GBA reports negative slack when PBA slack is positive, schedule and chip resources are wasted to fix false timing violations; when both GBA and PBA report negative slack, there is waste from from over-fixing per the GBA report; etc. Similar considerations apply to accuracy requirements for prediction of PBA slack itself. **(2) Prediction of timing at "missing corners".** Today's signoff timing analysis is performed at 200+ corners, and even P&R and optimization steps of physical design must satisfy constraints at dozens of corners. [23] [24] note that prediction of STA results for one or more "missing" corners that *are not* analyzed, based on the STA reports for corners that *are* analyzed, corresponds to *matrix completion* in ML [6] - and that the outlook for this ML application is promising. An implicit challenge is to identify *or synthesize* the K timing corners that will enable the most accurate prediction of timing at all N production timing corners. Product teams can also inform foundries and library teams of these K corners, so that the corresponding timing libraries can be the first to be characterized.

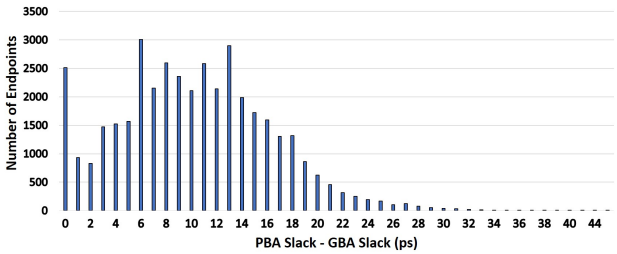

Figure 4: Frequency distribution of ((PBA slack) – (GBA slack)) at endpoints of *netcard*, 28FDSOI.

(3) Other analysis correlations. There are numerous other analysis correlation opportunities for ML. Often, these are linked with the prediction of tool and flow outcomes discussed below. Examples include correlation across various "multiphysics" analysis trajectories or loops [7] [22], such as those involving voltage droop or temperature effects in combination with normal signal integrity-aware timing. And, prominent among many parasitic estimation challenges is the prediction of bump inductance as early as possible in the die-package codesign process [22].

3 MODELS OF TOOLS AND DESIGNS

Convergent, high-quality design requires accurate modeling and prediction of downstream flow steps and outcomes. Predictive models (e.g., of wirelength, congestion, timing, etc.) become objectives or guides for optimizations, via a "modeling stack" that reaches up to system, architecture, and even project and enterprise levels. There is an urgent, complementary need for improved methods to (i) identify structural attributes of design instances that determine flow outcomes, (ii) identify "natural structure" in netlists (cf. [37]), and (iii) construct synthetic design proxies ("eye charts") [13][25][39] to help develop models of tools and flows. More broadly, tool and flow predictions are needed with increasing "span" across multiple design steps: the analogy is that we must predict what happens at the end of a longer and longer rope when the rope is wiggled.

Several examples of predictive models for tools and flows are reviewed in [23]. [8] demonstrates that learning-based models can accurately identify routing hotspots in detailed placement, and enable *model-guided optimization* whereby predicted routing hotspots are taken into account during physical synthesis with predictor-guided cell spreading. This addresses today's horrific divergence between global routing and final detailed routing, which stems from constraints on placement and pin access. Figure 5 [8] illustrates the discrepancy between routing hotspots (DRCs) predicted from global routing congestion, versus actual post-detailed routing DRCs. False positives in the former mislead routability optimizations and cause unnecessary iterations back to placement, while false negatives lead to doomed detailed routing runs. As with all other PD-related ML efforts thus far, the model of [8] incorporates parameters identified through domain expertise and multiple phases of model development. (Reducing this dependence could be a long-term goal for the field.) The work of [15] combines several simple predictions of layout and timing changes to predict clock buffer placement ECOs that will best improve clock skew variation across multiple timing corners. The work of [7] uses model parameters extracted from netlist, netlist sequential graph, floorplan, and constraints to predict post-P&R timing slacks at embedded memory instance endpoints. There are two clear takeaways from these experiences. First, there has been no escape from the need for deep domain knowledge and multiple, "highly curated" phases of model development. Second, results provide some optimism for the prospect of tool and flow prediction, based on models of both tools and design instances. The three reviewed works give a progression of "longer ropes": (i) from global/trial routing through detailed routing (and from ECO placement through incremental global/trial routing); (ii) from clock buffer and topology change through automated placement and routing ECOs, extraction, and timing analysis; and (iii) from netlist and floorplan information through placement, routing, optimization and IR drop-aware timing analysis.

Next Targets. [23] identifies two near-term targets for modeling of tools, flows and designs. **(1) Predicting doomed runs.** Substantial effort and schedule can be saved if a "doomed run" is avoided. Figure 6 shows four example progressions of the number of design rule violations during the (default) 20 iterations of a commercial router. Unsuccessful runs are those that end up with too many violations for manual fixing (e.g., the red and orange traces); these should be identified and terminated after as few iterations as possible. However, ultimately successful runs (e.g., the green trace) should be run to completion. Tool logfile data can be viewed as time series to which hidden Markov models [35] or policy iteration in Markov decision processes (MDPs) [4] may be applied. For the latter, collected logfiles from previous successful and unsuccessful tool runs can serve as the basis for automated extraction of a "blackjack strategy card" for a given tool, where "hit" analogizes to continuing the tool run for another iteration, and "stay" analogizes to terminating the tool run.[4] **(2) Feeding higher-level optimizations.** As noted above, predictive models must provide new objectives and guidance for higher-level optimizations. [1] points out that the scope for application extends up to project- and enterprise-level schedule and resource optimizations, with substantial returns possible.

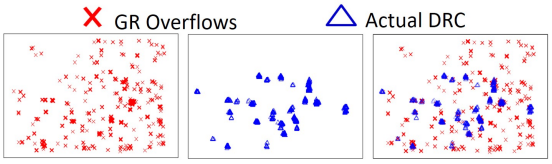

Figure 5: Post-route design rule violations (DRCs) predicted from global routing overflows (left); actual post-route DRCs (middle); overlay (right).

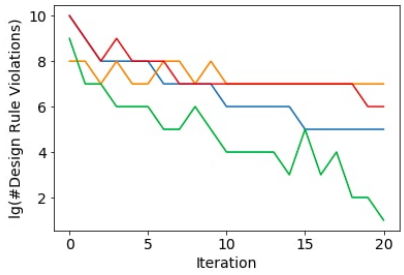

Figure 6: Four example progressions of the number of design rule violations (shown as a base-2 logarithm) with iterations of a commercial detailed router.

(3) Other Modeling and Prediction Needs. A first direction for future tool and flow modeling is to add confidence levels and probabilities to predictions. There is a trajectory of prediction from "can be achieved" to "will achieve" to "will achieve within X resources with Y probability distribution". A second direction is to improve the link between generation of data for model creation, and the model validation process. While physical design tools are not embedded in real-time, safety-critical contexts (i.e., impacts of poor modeling are likely limited to quality, cost and schedule), model accuracy must be as high as possible, as early as possible. Third, ML opportunities in physical design are clustered around "linchpin" flow steps: floorplan definition, logic synthesis, and handoff from placement to routing. For the logic synthesis step alone: since there is exactly one netlist handed off to implementation, what are the "magic" corners and constraints (including per-endpoint constraints [10]) that will induce the post-synthesis netlist that leads to best final implementation? Fourth, additional opportunities

[4] In the MDP paradigm, the *state space* used could consist of binned violation count and change in DRVs since a previous iteration; *actions* could be "go" or "stop", and *rewards* at each state used to derive the policy could include a small negative reward for a non-stop state, a large positive reward for termination with low number of DRVs, etc.

lie in finding the fixed point of a chicken-egg loop, as noted in [7] [22]. An example challenge today is to predict the fixed point for (non-uniform) power distribution and post-P&R layout that meets signoff constraints with maximum utilization.

4 SOC IMPLEMENTATION: A VISION

Physical design tools and flows today are unpredictable. A root cause is that many complex heuristics have been accreted upon previous complex heuristics. Thus, tools have become unpredictable, particularly when they are forced to *try hard*. Figure 7 (left), from implementation of the PULPino low-power RISC V core in a foundry 14nm enablement, shows that post-P&R area can change by 6% when target frequency changes by just 10MHz near the maximum achievable frequency. Figure 7 (right) illustrates that the statistics of this noisy tool behavior are Gaussian [32] [17]. Unpredictability of design implementation results in unpredictability of the design schedule. However, since product companies must strictly meet design and tapeout schedules, the design target (PPA) must be guardbanded, impacting product quality and profitability. Put another way: (i) our heuristics and tools are chaotic when designers demand best-quality results; and (ii) when designers want predictable results, they must aim low.

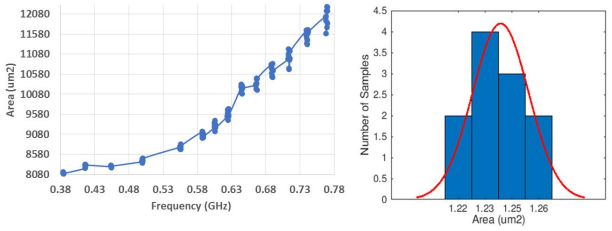

Figure 7: Left: SP&R implementation noise increases with target design quality. Right: Observed noise is essentially Gaussian.

SOC Design: Today. From Figure 7, a genesis of today's SOC physical implementation methodology can be seen, as illustrated in Figure 8(a). The figure illustrates that with unpredictable optimizers, as well as the perceived loss of "global optimization" of solution quality when the design problem is partitioned, designers demand as close to flat methodologies as possible. Hence, today's prevailing SOC methodology entails having as few large hard macros as possible. To satisfy this customer requirement in the face of Moore's-Law scaling of design complexity, EDA tools must add more heuristics so as to turn around ever-larger blocks in the same turnaround time. To recover design quality (e.g., in light of "aim low") designers seek as much flexibility as possible in their implementation tools.[5] This leads to poor predictability in design, which then leads to more iterations, and turnaround times become longer. Further, the lack of predictability induces larger design guardbands. As a result of these cause-effect relationships, the *achieved* design quality worsens, and the design capability gap grows. This is the unfortunate tale of coevolution between physical design tools and physical implementation methodology.

SOC Design: Future. To close the design capability gap, EDA and IC design together must "flip the arrows" of Figure 8(a). A vision for future SOC design is suggested in Figure 8(b). The physical implementation challenge is decomposed into many more small subproblems, by *hyperpartitioning* or "extreme partitioning"; this reduces the time needed to solve any given subproblem, and smaller subproblems can be better-solved (see [33]). At the same time, increasing the number of design partitions without undue loss of global solution quality demands new placement, global routing and optimization algorithms, as well as fundamentally new RTL partition and floorplan co-optimization capabilities. Further, *reducing* design flexibility by giving designers "freedoms from choice" with respect to RTL constructs, power distribution, clock distribution, global buffering, non-default wiring rules, etc. would increase predictability, leading to fewer iterations (ideally, single-pass design). Turnaround time is then minimized. Improved predictability and fewer iterations result in smaller design guardbands. The end result: improvement of *achieved* design quality, which shrinks the design capability gap. As pointed out in [24], achieving this vision of future SOC design methodology would improve quality, schedule and cost – i.e., "the last scaling levers". A number of new mindsets for tool developers and design flow engineers are implicit: (i) tools and flows should never return unexpected results; (ii) designers should see predictability, not chaos, in their tools and flows; (iii) cloud deployment and parallel search can help to preserve or improve achieved quality of results; and (iv) the focus of *design-based equivalent scaling* is on sustained reduction of design time and design effort.

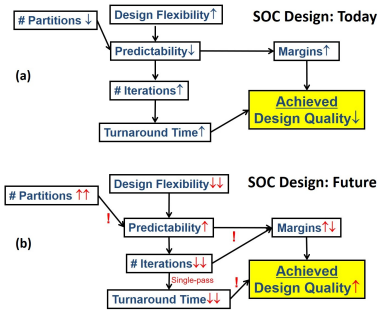

Figure 8: SOC design (a) today, and (b) in the future.

5 A ROADMAP FOR ML IN PD

This section describes a "roadmap" for the insertion of ML within and around physical design flow steps. Four high-level stages of insertion are described. Then, a list of specific, actionable challenges is given.

Four Stages of ML Insertion

Insertion of ML into and around physical design algorithms, tools and flows could be divided into four qualitatively distinct stages. Figure 9(a) conveys why IC implementation and design resource requirements are so challenging: there are thousands of potential options at each flow step (don't-use cells, timing constraints, pin placements, density screens, allowed netlist transforms, alternate commands-options and environment variables, ...), resulting in an enormous tree of possible flow trajectories. Today, even identifying a "best" among alternative post-synthesis netlists or physical floorplans to carry forward in the flow is beyond the grasp of human engineers. Thus, the likely **first stage** of ML insertion into IC will entail *creating robots*: mechanizing and automating (via expert systems, perhaps) 24/7 replacements for human engineers that reliably execute a given flow to completion.[6] Figure 10 shows

[5] A modern P&R tool has thousands of, and even more than ten thousand, command-option combinations.

[6] This goes beyond today's typical 'make chip' flow automation in that real expertise and human-seeming smarts are captured within the robot engineer. As discussed below, robots will likely also fill in last-mile or small-market tasks that are unserved by available tools.

how primitive "multi-armed bandit" (MAB) sampling can achieve resource-adaptive commercial synthesis, place and route with no human involvement – in a "robotic" manner that is distinct from expert systems approaches. Past tool run outcomes are used by the MAB to estimate the probability of meeting constraints at different parameter settings; future runs are then scheduled that are most likely to yield the best outcomes within the given (licenses × schedule) design resource budget. The figure shows the evolution of sampled frequencies versus iterations in the MAB's "robotic" execution.

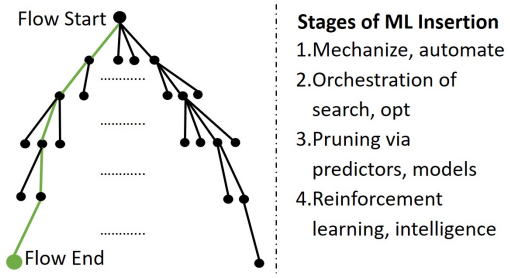

Figure 9: (a) Tree of options at flow steps. (b) Phases of ML insertion into production IC implementation.

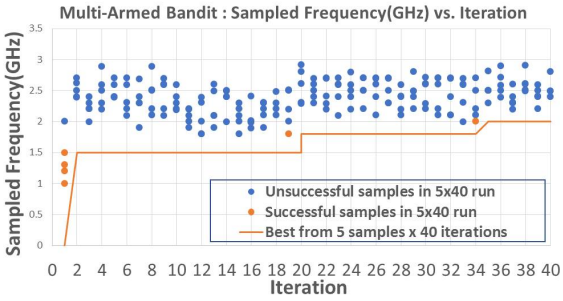

Figure 10: Trajectory of "no-human-in-the-loop" multi-armed bandit sampling of a commercial SP&R flow, with 40 iterations and 5 concurrent samples (tool runs) per iteration. Testcase: PULPino core in 14nm foundry technology, with given power and area constraints. Adapted from [21].

Once a robot engineer exists, the **second stage** of ML insertion is to optimally orchestrate N robot engineers that concurrently search multiple flow trajectories, where N can range from tens to thousands and is constrained chiefly by compute and license resources. Here, simple multistart, or depth-first or breadth-first traversal of the tree of flow options, is hopeless. Rather, it seems likely that strategies such as "go-with-the-winners" (GWTW) [2] will be applied. GWTW launches multiple optimization threads, and periodically identifies and clones the most promising thread while terminating other threads; see Figure 11(a). The GWTW method has been applied successfully in, e.g., [26]. Another promising direction may be *adaptive multistart* [5] [14], which exploits an inherent "big valley" structure in optimization cost landscapes to adaptively identify promising start configurations for iterative optimization. This is illustrated in Figure 11(b), where better start points for optimization are identified based on the structure of (locally-minimal) solutions found from previous start points.

The **third stage** will integrate *prediction* of tool- and design-specific outcomes over longer and longer subflows, so as to more surgically

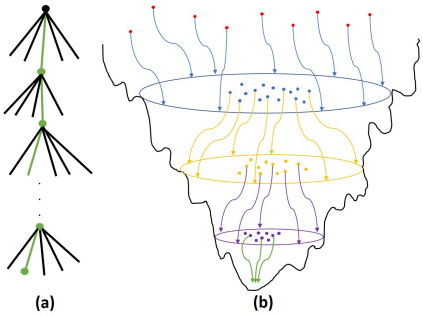

Figure 11: (a) Go-with-the-winners [2]. (b) Adaptive multi-start in a "big valley" optimization landscape [5] [14].

prune, terminate, or otherwise not waste design resources on less-promising flow trajectories. Implicit in the third stage is the improvement of *predictability* and modelability for PD heuristics and EDA tools. Finally, the **fourth stage** will span from reinforcement learning to "intelligence". At this stage, there are many obstacles. For example, the latency and unpredictability of IC design tool runs (we can't play the IC design game hundreds of millions of times in a few days, as we would the game of chess), the sparsity of data (there are millions of cat and dog faces on the web, but not many 10nm layouts), the lack of good evaluation functions, and the huge space of trajectories for design all look to be difficult challenges. Hopefully, aspects of the vision for future SOC design given above, and solutions to the initial challenges given below, will provide help toward realization of the fourth stage.

Specific Initial Challenges

Following are several specific "initial challenges" for machine learning in physical design.

"Last-Mile" Robots. A number of today's time-consuming, error-prone and even trial-and-error steps in IC implementation should be automated by systems that systematically search for tool command sequences, and/or observe and learn from humans. (1) *Automation of manual DRC violation fixing.* After routing and optimization, P&R tools leave DRC violations due to inability to handle latest foundry rules, unavoidable lack of routing resource in a high-utilization block, etc. PD engineers today must spread cells and perform rip-up and reroute manually. (2) *Automation of manual timing closure steps.* After routing and optimization, several thousand violations of maxtrans, setup and hold constraints may exist. PD engineers today fix these manually at the rate of several hundred per day per engineer. (3) *Placement of memory instances in a P&R block.* (4) *Package layout automation.* The ML challenge is to be able to assess the post-routed quality (e.g., with respect to bump inductances) of floorplan and pin map in die-package codesign. From this will flow bump/ball placement and placement improvement; a possible prerequisite is the automation of manual package routing.

Improving Analysis Correlation. (1) *Prediction of the worst PBA path.* For a given endpoint, the worst PBA path is not necessarily among any the top k GBA paths: CCS loads on side fanouts, path topology and composition, GBA common path pessimism removal, etc. all affect the rank correlation between GBA and PBA results of timing paths. (2) *Prediction of the worst PBA slack per endpoint, from GBA analysis.* E.g., from all GBA endpoint slacks. (3) *Prediction of timing at "missing corners".* Given timing analysis reports at k corners, predict reports at $N - k$ corners, where $k << N$. Similarly: given a prediction accuracy requirement, find $k << N$ corners, with k as small as possible, that enable prediction of remaining corners with the required accuracy. (4) *Closing of multiphysics analysis loops.*

I.e., as in [22] [7], with early priorities being vectorless dynamic IR drop and power-temperature loops. (5) *Continued improvement of timing correlation and estimation* as in [16] [30]. Matching the golden tool earlier in the flow will more accurately drive optimizations and reduce ECO iterations.

Predictive Models of Tools and Designs. (1) *Prediction of the convergent point for non-uniform PDN and P&R.* The PDN is defined before placement, but power analysis and routability impact can be assessed only after routing. (2) *Estimation of the PPA response of a given block in response to floorplan optimizations.* Final PPA impacts of feedthroughs, shape, utilization, memory placement, etc. must be comprehended to enable floorplan assessment and optimization (within a higher-level exploration of design partitioning/floorplanning solutions). (3) *Estimation of useful skew impact on post-route WNS, TNS metrics.* See, e.g., [10]. A low-level related challenge: predicting buffer locations to optimize both common paths and useful skew. (4) *"Auto-magic" determination of constraints for a given netlist*, for given performance and power targets – i.e., best settings for maxtrans, maxcap, clock uncertainty, etc. at each flow stage. More generally, determine "magic" corners and constraints that will produce the best netlist to send into P&R. (5) *Prediction of the best "target sequence" of constraints through layout optimization phases.* I.e., timing and power targets at synthesis, placement, etc. such that best final PPA metrics are achieved. (6) *Prediction of impacts (setup, hold slack, max transition, power) of an ECO, across MCMM scenarios.* (7) *Prediction of the "most-optimizable" cells during design closure.* Many optimization steps are wasted on instances that cannot be perturbed due to placement, timing, power and other context. (8) *Prediction of divergence (detouring, timing/slew violations) between trial/global route and final detailed route.* (9) *Prediction of "doomed runs"* at all steps of the physical design flow.

And More. (1) *Infrastructure for ML in IC design.* Standards for model encapsulation, model application, IP-preserving model sharing, etc. are yet to be developed. (2) *Standard ML platform for EDA modeling*. Enablement of design metrics collection, tool and flow model generation, design-adaptive tool and flow configuration, prediction of tool and flow outcomes, etc. would realize the original vision of METRICS [36] [11] [31]. (3) *Development of more modelable algorithms and tools* with smoother, less-chaotic outcomes than present methods. (4) *Development of datasets to support ML.* This spans new classes of artificial circuits and "eyecharts", as well as sharing of training data and the data generation task across different design organizations.

6 ACKNOWLEDGMENTS

Many thanks are due to Dr. Tuck-Boon Chan, Dr. Jiajia Li, Dr. Siddhartha Nath, Dr. Stefanus Mantik, Dr. Kambiz Samadi, Dr. Kwangok Jeong, Ms. Hyein Lee and Mr. Wei-Ting Jonas Chan who, along with current ABKGroup students and collaborators, performed much of the research cited in this paper. I thank Professor Lawrence Saul for ongoing discussions and collaborations. Permission of coauthors to reproduce figures from works referenced here is gratefully acknowledged. Research at UCSD is supported by NSF, Qualcomm, Samsung, NXP, Mentor Graphics and the C-DEN center.

REFERENCES

[1] P. Agrawal, M. Broxterman, B. Chatterjee, P. Cuevas, K. H. Hayashi, A. B. Kahng, P. K. Myana and S. Nath, "Optimal Scheduling and Allocation for IC Design Management and Cost Reduction", *ACM TODAES* 22(4) (2017), pp. 60:1-60:30.
[2] D. Aldous and U. Vazirani, "Go With the Winners", *Proc. IEEE Symp. on Foundations of Computer Science*, 1994, pp. 492-501.
[3] S. Bansal and R. Goering, "Making 20nm Design Challenges Manageable", http://www.chipdesignmag.com/pdfs/chip_design_special_DAC_issue_2012.pdf
[4] D. Bertsekas, *Dynamic Programming and Optimal Control*, Athena, 1995.
[5] K. D. Boese, A. B. Kahng and S. Muddu, "New Adaptive Multistart Techniques for Combinatorial Global Optimizations", *Operations Research Letters* 16(2) (1994), pp. 101-113.
[6] E. J. Candes and B. Recht, "Exact Matrix Completion via Convex Optimization", *Foundations of Computational Mathematics* 9 (2009), pp. 717-772.
[7] W.-T. J. Chan, K. Y. Chung, A. B. Kahng, N. D. MacDonald and S. Nath, "Learning-Based Prediction of Embedded Memory Timing Failures During Initial Floorplan Design", *Proc. ASP-DAC*, 2016, pp. 178-185.
[8] W.-T. J. Chan, P.-H. Ho, A. B. Kahng and P. Saxena, "Routability Optimization for Industrial Designs at Sub-14nm Process Nodes Using Machine Learning", *Proc. ISPD*, 2017, pp. 15-21.
[9] T.-B. Chan, A. B. Kahng, J. Li and S. Nath, "Optimization of Overdrive Signoff", *Proc. ASP-DAC*, 2013, pp. 344-349.
[10] T.-B. Chan, A. B. Kahng and J. Li, "NOLO: A No-Loop, Predictive Useful Skew Methodology for Improved Timing in IC Implementation", *Proc. ISQED*, 2014, pp. 504-509.
[11] S. Fenstermaker, D. George, A. B. Kahng, S. Mantik and B. Thielges, "METRICS: A System Architecture for Design Process Optimization", *Proc. DAC*, 2000, pp. 705-710.
[12] R. Goering, "What's Needed to "Fix" Timing Signoff?", *DAC Panel*, 2013.
[13] P. Gupta, A. B. Kahng, A. Kasibhatla and P. Sharma, "Eyecharts: Constructive Benchmarking of Gate Sizing Heuristics", *Proc. DAC*, 2010, pp. 597-602.
[14] L. Hagen and A. B. Kahng, "Combining Problem Reduction and Adaptive Multi-Start: A New Technique for Superior Iterative Partitioning", *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems* 16(7) (1997), pp. 709-717.
[15] K. Han, A. B. Kahng, J. Lee, J. Li and S. Nath, "A Global-Local Optimization Framework for Simultaneous Multi-Mode Multi-Corner Skew Variation Reduction", *Proc. DAC*, 2015, pp. 26:1-26:6.
[16] S. S. Han, A. B. Kahng, S. Nath and A. Vydyanathan, "A Deep Learning Methodology to Proliferate Golden Signoff Timing", *Proc. DATE*, 2014, pp. 260:1-260:6.
[17] K. Jeong and A. B. Kahng, "Methodology From Chaos in IC Implementation", *Proc. ISQED*, 2010, pp. 885-892.
[18] K. Jeong, A. B. Kahng and K. Samadi, "Impacts of Guardband Reduction on Design Process Outcomes: A Quantitative Approach", *IEEE Trans. Semiconductor Manufacturing* 22(4) (2009), pp. 552-565.
[19] A. B. Kahng, "The Cost of Design", *IEEE Design & Test of Computers*, 2002.
[20] A. B. Kahng, "The ITRS Design Technology and System Drivers Roadmap: Process and Status", *Proc. DAC*, 2013, pp. 34-39.
[21] A. B. Kahng, DARPA IDEA Workshop presentation, Arlington, April 2017.
[22] A. B. Kahng, ANSYS Executive Breakfast keynote talk, June 2017. http://vlsicad.ucsd.edu/Presentations/talk/Kahng-ANSYS-DACBreakfast_talk_DISTRIBUTED2.pdf
[23] A. B. Kahng, "New Directions for Learning-Based IC Design Tools and Methodologies", *Proc. ASP-DAC*, 2018, pp. 405-410.
[24] A. B. Kahng, "Quality, Schedule, and Cost: Design Technology and the Last Semiconductor Scaling Levers". *keynote talk, ASP-DAC*, 2018. http://vlsicad.ucsd.edu/ASPDAC18/ASP-DAC-2018-Keynote-Kahng-POSTED.pptx
[25] A. B. Kahng and S. Kang, "Construction of Realistic Gate Sizing Benchmarks With Known Optimal Solutions", *Proc. ISPD*, 2012, pp. 153-160.
[26] A. B. Kahng, S. Kang, H. Lee, I. L. Markov and P. Thapar, "High-Performance Gate Sizing with a Signoff Timer", *Proc. ICCAD*, 2013, pp. 450-457.
[27] A. B. Kahng, S. Kang, H. Lee, S. Nath and J. Wadhwani, "Learning-Based Approximation of Interconnect Delay and Slew in Signoff Timing Tools", *Proc. SLIP*, 2013, pp. 1-8.
[28] A. B. Kahng, B. Lin and S. Nath, "Enhanced Metamodeling Techniques for High-Dimensional IC Design Estimation Problems", *Proc. DATE*, 2013, pp. 1861-1866.
[29] A. B. Kahng, B. Lin and S. Nath, "High-Dimensional Metamodeling for Prediction of Clock Tree Synthesis Outcomes", *Proc. SLIP*, 2013, pp. 1-7.
[30] A. B. Kahng, M. Luo and S. Nath, "SI for Free: Machine Learning of Interconnect Coupling Delay and Transition Effects", *Proc. SLIP*, 2015, pp. 1-8.
[31] A. B. Kahng and S. Mantik, "A System for Automatic Recording and Prediction of Design Quality Metrics", *Proc. ISQED*, 2001, pp. 81-86.
[32] A. Kahng and S. Mantik, "Measurement of Inherent Noise in EDA Tools", *Proc. ISQED*, 2002, pp. 206-211.
[33] A. Katsioulas, S. Chow, J. Avidan and D. Fotakis, "Integrated Circuit Architecture with Standard Blocks"', *U.S. Patent* 6,467,074, 2002.
[34] C. W. Moon, P. Gupta, P. J. Donehue and A. B. Kahng, "Method of Designing a Digital Circuit by Correlating Different Static Timing Analyzers", *US Patent* 7,823,098, 2010.
[35] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications on Speech Recognition", *Proc. IEEE* 77 (1989), pp. 257-286.
[36] The GSRC METRICS Initiative. http://vlsicad.ucsd.edu/GSRC/metrics/
[37] Partitioning- and Placement-based Intrinsic Rent Parameter Evaluation. http://vlsicad.ucsd.edu/WLD/RentCon.pdf
[38] "DARPA Rolls Out Electronics Resurgence Initiative", https://www.darpa.mil/news-events/2017-09-13
[39] Gate Sizing Benchmarks With Known Optimal Solution. http://vlsicad.ucsd.edu/SIZING/bench/artificial.html
[40] *International Technology Roadmap for Semiconductors.* http://www.itrs2.net/itrs-reports.html