# Machine Learning for CAD/EDA: The Road Ahead

**Andrew B. Kahng**
Computer Science and Engineering Department
Electrical and Computer Engineering Department
University of California at San Diego
La Jolla, CA 92093 USA

*Editor's notes:*
This keynote article sketches out a vision for the development of machine learning (ML) for computer-aided design (CAD): from today's ML-assisted CAD (MLCAD) to tomorrow's ML-based design automation (MLDA).
*—Ulf Schlichtmann, Technical University of Munich*

**THE ROAD AHEAD** for machine learning (ML) and computer-aided design (CAD)/EDA is built from three elements—learning, optimization, and CAD itself. *Learning* is the improvement of a computer agent's perception, knowledge, or actions based on experience or data [1]. *Optimization* is the universal quest to do better: it is a centuries-old discipline that is at the heart of leading-edge IC design. CAD is our world: a high-stakes use domain for learning and optimization that brings staggering scale and complexity along with multiple abstractions and objectives. Learning, optimization, and CAD are united in service of *scaling*, which is the realization of more value while consuming fewer resources (energy, time, area, and cost). Scaling makes the impossible possible: it propels IC CAD/EDA, IC design, and the broader semiconductor ecosystem forward into the future.

The IC design optimization problem has essentially unbounded complexity. As illustrated in Figure 1, there is a starting point for design, with inputs such as register-transfer level Verilog and constraints. Then, there are many steps and decision points (e.g., test insertion, retiming, and leakage optimization effort) on the way to a complete outcome (e.g., postroute layout and signoff reports) at a flow end leaf. From start to end is expensive, requiring weeks to months of effort. The goal is to achieve the best possible outcome, but there is an enormous space of trajectories, and the design optimization must stay within a given "*box*" *of resources*: 1) servers; 2) licenses; 3) people; and 4) schedule. There are never enough resources for optimization.

With the slowdown of device and process scaling, more burden is placed on design technology-based "equivalent scaling" to improve IC product quality, development schedule, and cost. Here, ML offers important boosters to CAD/EDA. First, ML enables *prediction*: seeing what lies ahead in the design process. ML models provide predictions that can be leveraged in design exploration, while also serving as objectives for higher-level optimizations. What we cannot predict, we must guardband—and what we do not have time to explore, we leave on the table.

Second, ML enables *optimization*, not only helping to solve difficult optimizations, but also giving new perspectives on classic optimization formulations. Frameworks such as learning to optimize [2], graphical neural network (GNN)-based embedding [3], and reinforcement learning (RL) [4] lie on the road from today's ML-assisted CAD (MLCAD) to

*Digital Object Identifier 10.1109/MDAT.2022.3161593*
*Date of publication: 13 July 2022; date of current version: 20 January 2023.*



Copublished by the IEEE CEDA, IEEE CASS, IEEE SSCS, and TTTC

tomorrow's ML-based design automation (MLDA): intelligent, self-driving CAD/EDA tools and flows.

Third, ML is effectively deployed on *modern compute resources* such as cloud and GPU. This provides new paths to scalable CAD/EDA optimizations that can exploit massive data and numbers of threads. This is a crucial aspect of what we might think of as a future "EDA2.0." Here, relevant frameworks include federated and evolutionary methods, stochastic gradient descent, sampling, and gradient-free optimization.

## ML and prediction

Within the design process, models and predictors are essential for cost and schedule reduction, as well as for the quality of results (QoR) improvement. ML-based predictions of delay, slew, coupling, and other parameters can shift the cost-accuracy tradeoff curve for electrical analysis, as illustrated in Figure 2 [5]. Reducing uncertainty means less guardbanding and increased QoR. Analogously, prediction of eventual quality metrics or tool failures enables early pruning of "doomed runs," allowing optimization resources to be better spent. Predicting farther ahead, with greater accuracy, affords more leverage. (The flip side of this is the development of more predictable and modelable heuristics and tools.)

Quality metrics such as clock skew or pin access failures in detailed routing become easier to predict as a design progresses from RTL to the netlist, floorplan, global placement, and onward, that is, as more information becomes available. Eventually, such metrics become frozen (e.g., after detailed routing) and trivial to "predict." In this light, ML-based predictions during design implementation also shift a tradeoff curve, namely, one of accuracy versus available information. The choice of tradeoff point (i.e., of error or risk versus wall time) is an important hyperparameter in sampling and distributed learning for optimization.

ML for modeling and prediction in CAD/EDA must surmount several basic challenges. First, the predictability of today's metaheuristic optimization outcomes decreases as solution quality increases. In other words, our heuristics and tools become noisier and "chaotic" when pushed to their limits, which unfortunately is the regime in which IC design needs the most help. Future mitigations may include more predictable heuristics, predictors of distributions and order statistics of ensembles of runs, and improved sampling methods. Better predictors will also explain more of the variability within the design process, just as manufacturing variations have moved from "random" to "systematic" with improved understanding.

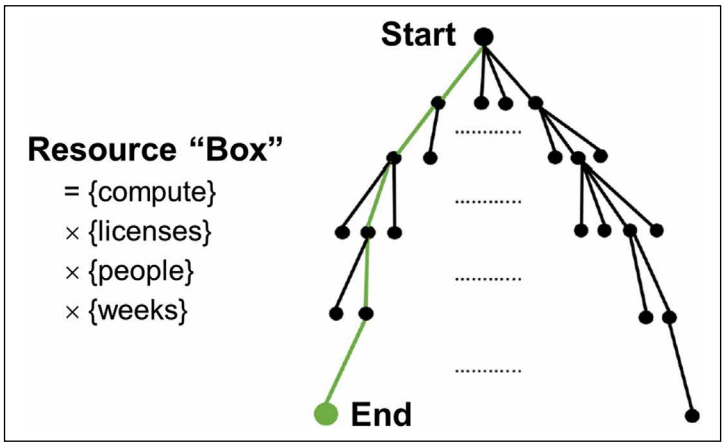


**Figure 1. A huge search space of options is implicit in design optimization. The optimization itself must live in a "box" of resources.**

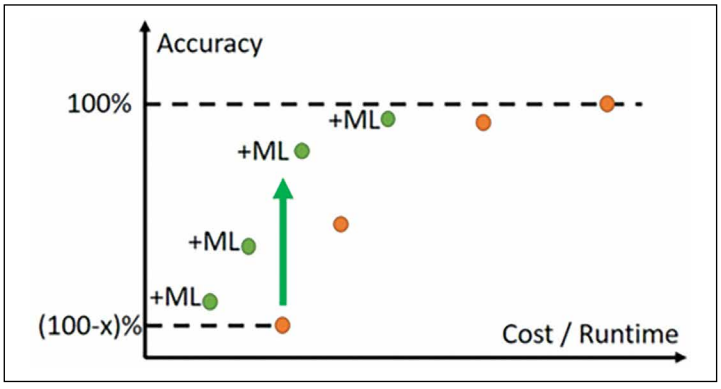


**Figure 2. ML shifts the accuracy-versus-cost tradeoff curve in electrical analyses. ML-based predictions can similarly shift the tradeoff of accuracy versus information regime in design implementation.**

Second, IC design optimizations often bring "max" (as opposed to "sum") objectives and cost landscapes in which useful gradient information is difficult to discern. It is the longest timing path or the worst routing hotspot that must be estimated and optimized. Structurally dissimilar solutions can have similar quality metrics such as worst timing slack or wirelength, but *which* paths are critical or *which* layout regions are congested can differ greatly between these solutions.

Third, what is predicted—and how predictive models will be used—needs careful consideration. For example, predicting achievable solution quality

(e.g., maximum clock frequency) is relatively useless without a "certificate," namely, a flow setup and runscript that will actually achieve such a solution. Even perfect predictions can be difficult or even harmful when we try to use them, as noted in [6]. This is because when we predict an outcome, such as the total area of buffers that will be inserted in a netlist or the final placed location of a macro-block, acting on the prediction will change the predicted outcome. Hence, "Be careful what you ask for from ML (and, how you will use it)."

## ML and optimization

Recent years have shown that ML can help to solve difficult CAD/EDA optimizations via predictive modeling, efficient sampling, hyperparameter tuning, RL, and many other means. Conversely, optimization is a fundamental component of modern ML, whether in stochastic gradient descent, sampling, or distributed contexts. (At a meta-level, CAD/EDA optimizations improve the hardware systems on which learning takes place.) The future scaling of CAD/EDA technology will build on the close interplay between learning and optimization, as well as the representations that unite them via domain knowledge and mathematical structure. Example research foci include: 1) the interaction between discrete-combinatorial and continuous methods; 2) optimization and sampling on manifolds; 3) sequential decision-making; 4) nonconvex optimization and deep learning; and 5) distributed and federated learning and optimization [7]. Observations regarding several near-term prospects are as follows.

### Embedding and scaling

First, CAD/EDA has always contended with scale through problem size reductions, notably by partitioning and clustering of (embedded) graph or hypergraph representations. In recent years, large-scale (message-passing) GNNs have shown promising applications to combinatorial optimization and algorithm alignment [2], [3], [8]. GNN-based methods, with attention and transformers, offer the promise of new optimizations and solution quality beyond what human experts can achieve [4], as well as clustering and embedding that more naturally discovers important problem structures without hand-crafting of representations and hyperparameters. Orthogonally (see the upcoming section), several challenges posed by sparse and/or confidential data have been met by the use of layered model architectures along with transfer learning and few-shot learning. There are also longer-range challenges, particularly for combinatorial optimizations. These include solution quality, generalization, and mechanisms for diagnosis and debugging of models.

### From AI-boosted to AI-based

Second, making tools and flows "smarter" has been an active area for ML application, with well-documented benefits. Supervised learning and intelligent sampling of hyperparameter spaces have been used to develop smart flows that are inherently tuned to task distributions. Autotuning methods use sampling and estimation of distributions to perform sequential black-box hyperparameter optimization. Such methods balance exploration and exploitation within optimization resource constraints and have important adjacencies in stochastic and evolutionary optimization. On the other hand, smart flows and autotuning still remain at a "knob-twiddling" level, following whatever structure happens to be imposed by existing CAD/EDA optimization tools. This largely ignores expert designer knowledge, as well as the stack of models and symbolic representations that underlie chip design. A longer-range challenge is to marry data-centric and knowledge-centric approaches to achieve AI-based optimization and intelligent design flows, in the sense of "third-wave AI."

### Lessons of "The Bitter Lesson"

Third, it is timely to revisit "the bitter lesson" [9], which is that "general methods that leverage computation are ultimately the most effective, and by a large margin." As witnessed by the histories of chess, Go, and speech recognition, the general methods that have scaled well with available computation are search and learning. Theoretical advances such as the double descent risk curve of Belkin et al. [10] explain the success of large, overparameterized neural-network models that operate beyond an "interpolation threshold." Recently, Mirhoseini et al. [4] describe a deep RL approach to chip floorplanning that yields solutions "superior or comparable to those produced by humans in all key metrics"; this is potentially another instance of the bitter lesson, which CAD/EDA and IC design teams are seeking to reproduce and translate across many contexts.

Corollaries of the bitter lesson, while discomfiting, might also give us encouragement. In our lifetimes,

available computation via process, circuit, and system innovations has scaled by only several tens of orders of magnitude, while the state or solution spaces for design optimization have grown by much larger factors. We might ask whether "the bitter lesson" means that expert humans are not that hard to beat, and/or whether (and why) CAD/EDA problems are not so difficult, after all.

## Mind the (suboptimality) gap

As noted in [11], the reality of optimization is "better, faster, cheaper—pick any two." Somewhat curiously, in our field, we often insist (as a customer to an EDA supplier, or as an academic peer reviewer to an author with a new heuristic) on "I want all three." PhD students are trained to formulate and attack difficult optimizations using integer-linear programs, minimum-cost flows, primal-dual methods, dynamic programming, satisfiability, and so on. But in practice, "Our customers need an answer overnight," "The approach is impractical due to its runtime," or "While the method improves wirelength by 1%, this is not a fair comparison because runtimes are three times longer." Over the past decades, such messages have driven the CAD/EDA field to cut corners and add more heuristics on top of existing stacks of heuristics. This has come at a cost. Arguably, we are as ill-informed about suboptimality gaps for classical EDA optimizations and about the potential benefits of the longer running and/or distributed CAD optimizations, as we were 20+ years ago. However, in today's era of optimization, 1% matters.

In Figure 3a, heuristic B clearly dominates heuristic A. Heuristic B also seems to dominate heuristic C (e.g., with effort = $t_1$) until the computational resource is expanded (effort = $t_2$), and C pulls ahead. Thus, B and C *together* define the quality-effort Pareto. It is unfortunate when heuristic C never sees the light of day, particularly since we have little idea how close any of these heuristics are to reaching optimality.

Note, too, that quality and effort cannot be separated from each other in the Pareto frontier of optimization. Figure 3b and c illustrates aspects of how EDA optimization must live with this inseparability. Figure 3b illustrates the "noise" seen in today's CAD/EDA tools. This highlights the importance of sampling to find the good tail of distributions. A key aspect of this is that even sampling needs to become much smarter, for example, to decide how much sampling is needed before giving up on a QoR target as "not

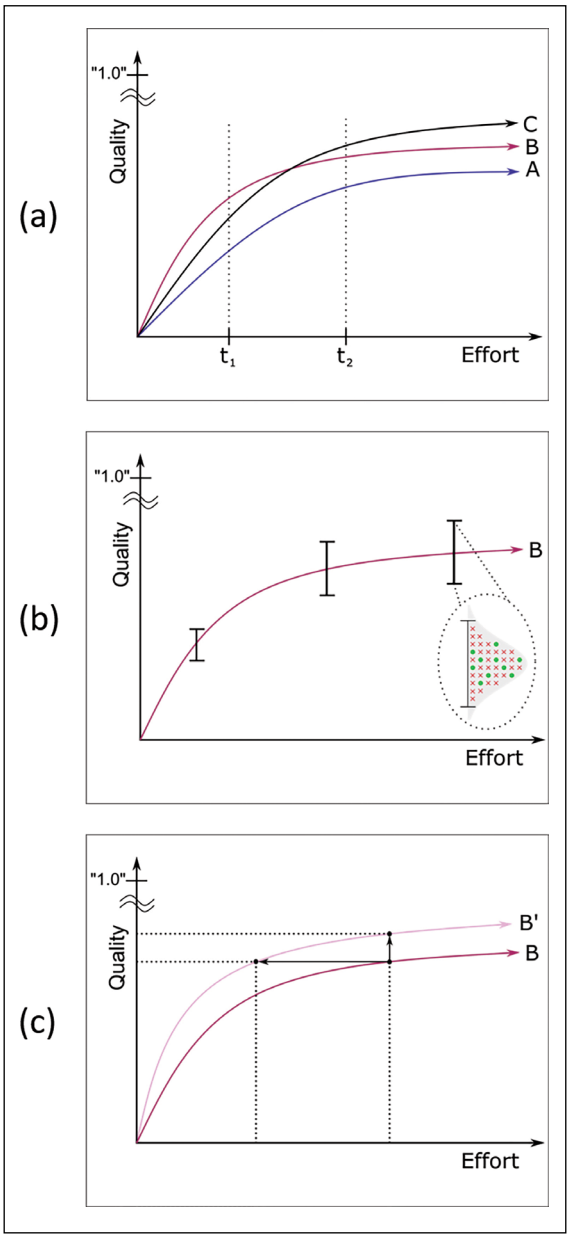


**Figure 3. (a) Multiple approaches together define the quality-versus-effort Pareto frontier. For many optimizations, the suboptimality gap is unknown. In the figure, quality = "1.0" denotes optimality (in optimization) or accuracy (in analysis). (b) At the limits of achievable QoR, heuristic behavior becomes less smooth, and the density of feasible solutions can decrease even while remaining nonzero. (c) ML can improve QoR achieved with a given effort (resource budget), and/or reduce the effort needed to achieve a given QoR. See the "Inset" for two case studies.**

possible." The figure illustrates how aiming for a given design QoR target can result in a mix of red fails and green successes within the distribution of QoR outcomes. Figure 3c shows how rearchitecting tools and ML-based optimization methods will "shift left and up," moving the state of CAD/EDA technology from B to B'. See the accompanying "Inset" for two case studies.

## A road starts with a roadbed

At the nexus of learning, optimization, and CAD, several foundational elements provide a "roadbed" for the road ahead. These include: 1) benchmarking and roadmapping of CAD/EDA optimizations; 2) data to enable data-driven methods; and 3) "EDA 2.0" that broadly reinvents core optimization algorithms and tool architectures for scalability on modern compute substrates.

### Benchmarking and roadmapping

First, CAD/EDA optimization is a *technology*, and as with any technology, its progress must be measured, benchmarked, and roadmapped. Indeed, DA has always been included as a key supplier of technology in the semiconductor industry's technology roadmap. At the same time, benchmarking has long been a fraught topic in CAD/EDA. Public benchmarks, even when "real," are obfuscated (e.g., module hierarchy stripped), incomplete (no clock), nonvertical (usable only for a specific academic contest), and past any competitive relevance (old). Moreover, no independent, trusted entity exists to assess EDA capability. An "Underwriters Laboratories for EDA" that can measure and benchmark IC design tools and flows would be a welcome development.

Benchmarking is naturally complemented by the roadmapping of CAD/EDA optimizations according to metrics such as the solution quality achieved within a given resource bound, or the resource needed to achieve a given solution quality. A strong culture of benchmarking, calibration, and "measure to improve" goes hand in hand with instrumentation and data collection, transparency, and reproducibility. This is reflected in recent (robust design flow, calibrations, and Metrics4ML) activities of the IEEE CEDA DA Technical Committee [12].

With real benchmarks, optimal solution costs, and remaining suboptimality gaps are unknown. Thus, artificial benchmarks are needed that can balance realism, known optimal solution quality, scalability, and other desiderata. Real designs can be perturbed into artificial test cases with known achievable solution costs or scaled to yield bootstrapped lower bounds on heuristic suboptimality. Alternatively, optimal solutions can be "planted" in artificial test cases, enabling suboptimality to be exactly measured. Several approaches produce artificial netlists that try to match prescribed degree distributions, I/O counts, path depths, sequential/combinational instance counts, and other criteria, which is in itself a difficult optimization.

### Data

Second, *data* is essential to ML for CAD/EDA. That is, ML enables optimization methods to be conceived from a *data-driven*, rather than a CS-theoretical, perspective [2]. Unfortunately, in our field data is sparse, closely guarded, and in constant flux with the evolution of architectures, standards, and technologies. When paths to real data are infeasible, research enablement requires "data virtual reality": data that is artificial yet scalable, free from biases, and indistinguishable from real data *from the perspective of optimization*. This brings numerous research challenges such as matching the behavior of real circuits through multistage optimizations (e.g., logic transforms, physical embedding, timing closure, and sizing) that span multiple abstractions. Jumping-off points include (graphical) generative adversarial network models and (federated) differential privacy techniques, in conjunction with obfuscation and noising—as well as the finding of relevant topological and other motifs that characterize real designs. Data augmentation, along with transfer learning and low-shot learning, can further mitigate the unavailability of proprietary data. New techniques will also be needed to *project* data—from process design kits (PDKs) and libraries to system designs—forward into future technologies. This is because optimizations that meet the leading edge of practice cannot be driven only by the rearview mirror.

### Computing platforms

Third, reinventing CAD/EDA to leverage *modern computational resources* is also a mandatory aspect of closing the suboptimality gap. Since the 1980s, CAD/EDA research and development has focused on single-threaded or single-server turnaround times. However, future solution quality and turnaround time gains must leverage the hyperscaling

of available compute on platforms that span cloud, GPU, and accelerator hardware. Core EDA algorithms and architectures will need to be reinvented accordingly.

As researchers make foundational advances in (sum-of-squares, manifold, nonconvex, min–max, etc.) optimization, distributable algorithm realizations and distributed-federated learning and optimization will also be needed. Basic open questions surround the tension between exploration and exploitation, and the need to deliver "anytime" solution quality that improves monotonically with the given footprint (threads × runtime) of the computational resource. For example, what new principles will improve our understanding of how much additional compute is needed to achieve a prescribed wall time reduction, iso-QoR? Or, how much additional compute and/or wall time is needed to achieve a prescribed increase in QoR? From particle swarm optimization to differential evolution and the class of EDAs for EDA!, there is a rich body of work on metaheuristics and "parallel problem-solving from nature" that can potentially contribute to the development of future cloud-scalable (and learning-enabled) CAD/EDA.

We build roads as infrastructure so that we can go farther and go faster. How far and how fast we can travel along the road ahead for ML and CAD/EDA is yet to be seen. There are difficult technical challenges, such as debugging of models, marrying of data- and knowledge-driven AI, end-to-end direct inference of design solutions, and comprehension of security as an optimization objective. Progress will also depend on business models, investments in basic research, minimization of redundant efforts, and other nontechnical factors. And, without a roadbed, there can be no road: benchmarking, roadmapping, data, and the use of modern compute platforms are all essential to the scaling of ML-enabled EDA optimization. Last, if there are too many toll booths and speed bumps (closed ecosystems, absence of standards, assertions of IP, etc.), then there will be fewer travelers on the road ahead.

At the same time, as shown in Figure 4, the road ahead for ML and CAD/EDA is a highway with many lanes [13]. It leads to self-driving IC design tools and flows that make design innovation and solution space exploration more effective, efficient, and accessible. It also takes us from today's ML-assisted CAD (MLCAD) to tomorrow's intelligent MLDA. At every step of the way, ML will help EDA tools and flows extract more from optimization resources, enabling more iterations, and exploration within the design schedule. Building this road and driving along it will unite learning, optimization, and CAD—and stakeholders across the design and EDA ecosystem as well—to bring future scaling benefits to the industry and society at large.

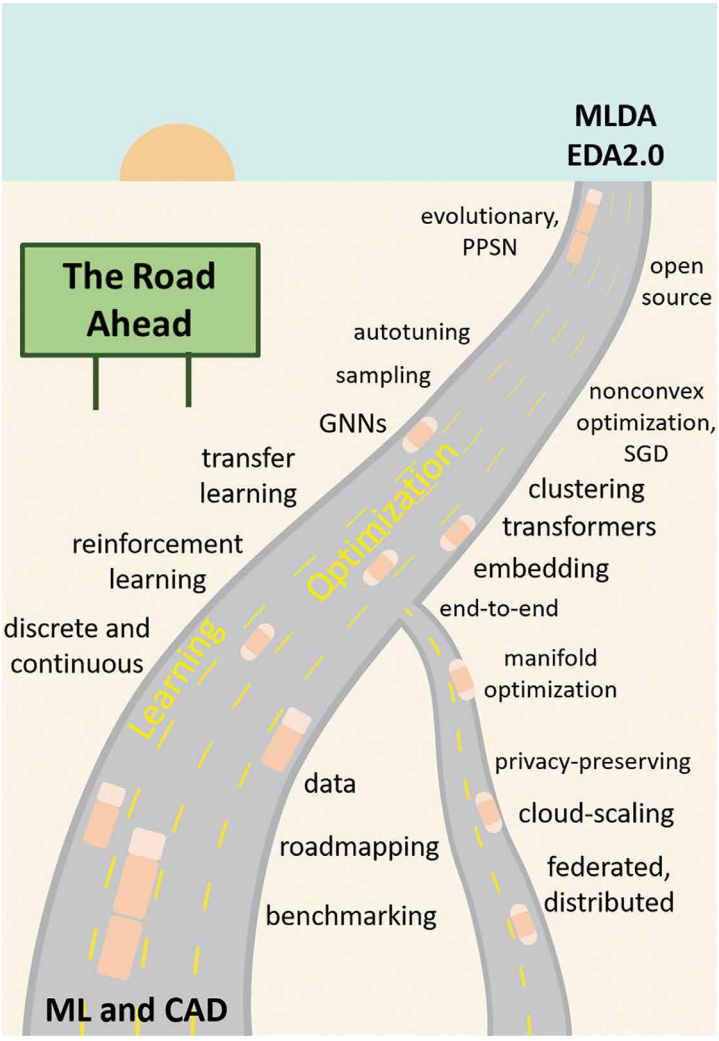


**Figure 4. Road ahead for ML and CAD/EDA is a highway with many lanes and many travelers.**

## Acknowledgments

Many thanks are due to the Guest Editors for their invitation to write this article, which builds on [14]. Thanks are also due to Leon Stok, Gi-Joon Nam, and IBM colleagues, as well as to many collaborators in the TILOS AI institute [7], for inspiring discussions and shared visions of the road ahead.

**Inset**

The challenges illustrated in Figure 3b can be seen in the following case study using a leading commercial place-and-route tool. Figure 5 contrasts what is requested from the tool, versus what is delivered. The *x*-axis gives the target clock period (CP) of the design, while the *y*-axis gives the effective, that is, actual, CP of the place-and-route outcome.

The *x*-axis has 23 target CP values, spaced 50 ps apart. For each, 101 distinct tool runs are made, all aiming within half a picosecond of that target; this is achieved by stepping the CP target by 0.01 ps within a 1 ps range. How to obtain the best-possible (e.g., minimum-CP) outcome within a given budget of samples is an open challenge for MLCAD. Methods are needed to characterize and sample from the transition between easy CP targets on the left, and impossible targets on the right—comprehending the probability of functional failures (red points) and the mapping between target and effective CP values.

Next, note that the "shift left and up" illustrated in Figure 3c has significant potential value in, for example, the detailed routing stage of physical design. The challenge is to identify "doomed runs" and to intelligently reassign or repurpose their CPU resources, such that overall success is more likely, and overall turnaround time is reduced.

Here, a second illuminating case study involves detailed routing. Modern detailed routing tools partition the routing task into disjoint "tiles" or "switchboxes," assigning a worker thread to each tile. After all tiles have been processed, those that do not yet have a clean routing solution are reprocessed—for example, with using a variant costing or connection ordering strategy—in the next iteration of the router. Ideally, the number of unsolved

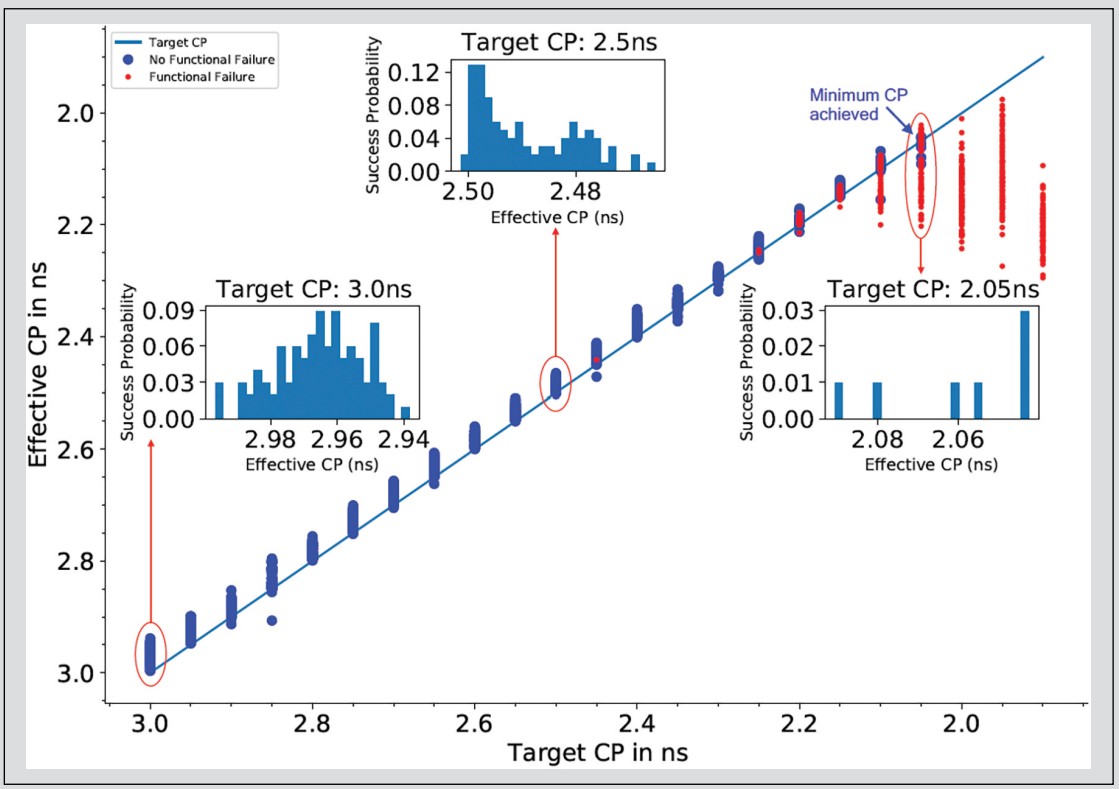


**Figure 5. Mapping between desired (target) CP and actual (effective) CP achieved by a commercial place-and-route tool. At each *x*-axis value, results are obtained for 101 target CP values within a 1 ps range of the given target CP. The highest blue data point in the plot corresponds to the minimum CP achieved, that is, the maximum-frequency result.**

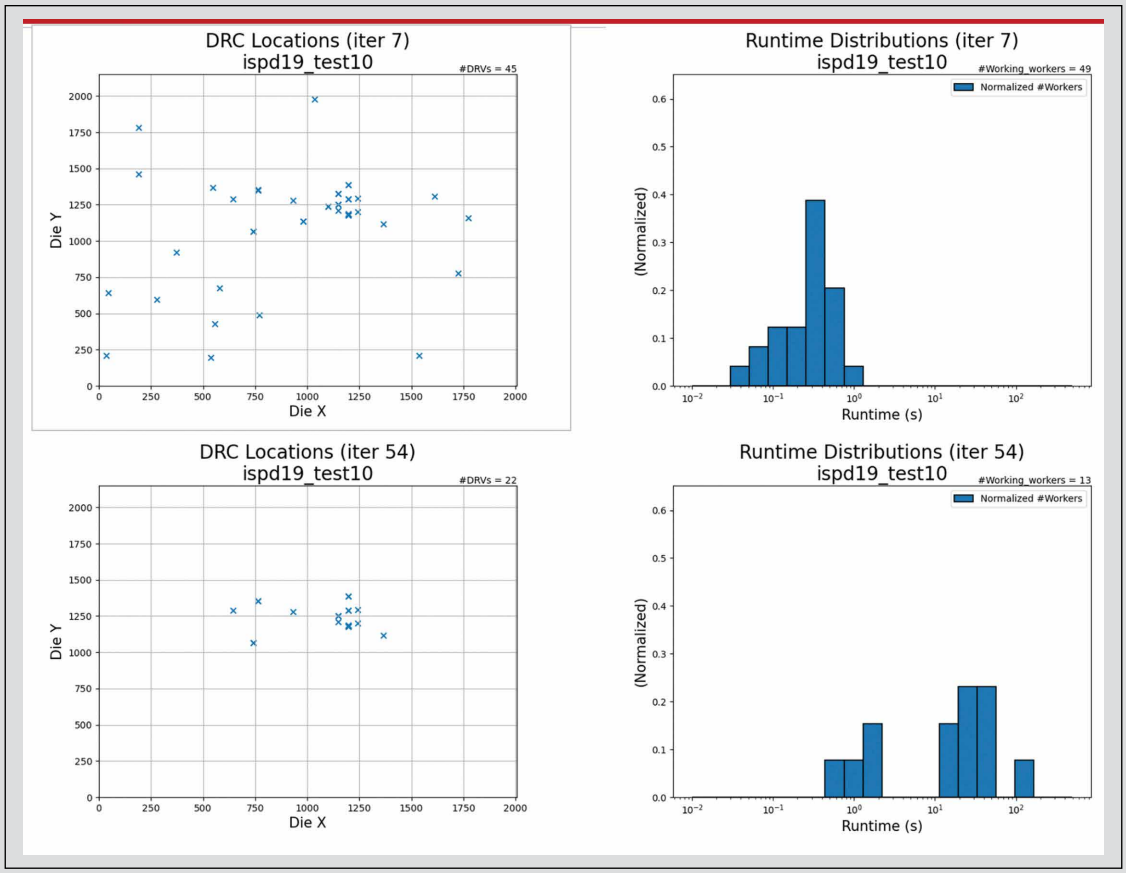


**Figure 6. Locations of detailed routing tiles (workers) with DRC violations seen at two iterations (7 and 54) of a leading academic routing tool (left). Distribution of runtimes, shown in seconds on a logarithmic scale, for the workers (right). Stubborn routing tiles could receive more attention (more assigned workers, executing variant solution strategies) if identified earlier in the routing process.**

design rule check violations (DRCs) and remaining tiles will decrease with each iteration of the router.

Figure 6 shows the location of tiles that have remaining DRC violations, as well as runtimes per worker, at two iterations of a leading academic routing tool. (Note the logarithmic scale used for runtimes.) In early iterations, average runtimes per worker are a fraction of a second, but in later iterations some workers can run for hundreds of seconds. This offers high-value opportunities for learning (e.g., predicting "doomed" runs or where stubborn DRC violations will occur) as well as optimization (e.g., adaptive strategies to resolve DRCs in a given tile, and budgeting of available worker threads to remaining tiles).

## References


[1] C. Manning. (Sep. 2020). *Artificial Intelligence Definitions*. [Online]. Available: https://hai.stanford.edu/sites/default/files/2020-09/AI-Definitions-HAI.pdf

[2] T. Chen et al., "Learning to optimize: A primer and a benchmark," 2021, *arXiv:2103.12828*.

[3] Q. Cappart et al., "Combinatorial optimization and reasoning with graph neural networks," 2021, *arXiv:2102.09544*.

[4] A. Mirhoseini et al., "A graph placement methodology for fast chip design," *Nature*, vol. 594, no. 10, p. 207, Jun. 2021.

[5] A. B. Kahng, "Machine learning applications in physical design: Recent results and directions," in *Proc. ISPD*, 2018, pp. 68–73.

[6] T.-B. Chan, A. B. Kahng, and M. Woo, "Revisiting inherent noise floors for interconnect prediction," in *Proc. ACM/IEEE Int. Workshop Syst.-Level Interconnect Problems Pathfinding*, Nov. 2020, pp. 1–7.

[7] The Institute for Learning-Enabled Optimization at Scale. Accessed: Jul. 10, 2021. [Online]. Available: https://tilos.ai/

[8] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: A methodological tour d'horizon," *Eur. J. Oper. Res.*, vol. 290, no. 2, pp. 405–421, Apr. 2021.

[9] R. Sutton. (Mar. 2019). *The Bitter Lesson*. [Online]. Available: http://www.incompleteideas.net/IncIdeas/BitterLesson.html

[10] M. Belkin et al., "Reconciling modern machine learning practice and the bias-variance trade-off," 2018, *arXiv:1812.11118*.

[11] A. B. Kahng, "Advancing placement," in *Proc. ISPD*, 2021, pp. 15–22.

[12] IEEE CEDA Design Automation Technical Committee. *GitHub Repository*. Accessed: Jul. 10, 2021. [Online]. Available: https://github.com/ieee-ceda-datc

[13] G. Huang et al., "Machine learning for electronic design automation: A survey," 2021, *arXiv:2102.03357*.

[14] A. B. Kahng, "MLCAD today and tomorrow: Learning, optimization and scaling," in *Proc. MLCAD Workshop*, Nov. 2020, p. 1. [Online]. Available: https://www.youtube.com/watch?v=oVF3yUhyc

**Andrew B. Kahng** is a Distinguished Professor of computer science and engineering and electrical and computer engineering at the University of California San Diego (UC San Diego), La Jolla, CA 92093 USA. His research interests include IC physical design, design for manufacturability (DFM), technology roadmapping, and machine learning (ML) for EDA. Kahng has a PhD in computer science from UC San Diego. He is a Fellow of IEEE and ACM.

■ Direct questions and comments about this article to Andrew B. Kahng, Computer Science and Engineering Department and Electrical and Computer Engineering Department, University of California San Diego (UC San Diego) La Jolla, CA 92093 USA; abk@ucsd.edu.