AENEID: A Generic Lithography-Friendly Detailed Router Based on Post-RET Data Learning and Hotspot Detection

Duo Ding, Jhih-Rong Gao, Kun Yuan and David Z. Pan
ECE Dept. The Univ. of Texas at Austin, Austin, TX 78712
{ding, jrgao, kyuan, dpan}@cerc.utexas.edu

ABSTRACT

In the era of deep sub-wavelength lithography for nanometer VLSI designs, manufacturability and yield issues are critical and need to be addressed during the key physical design implementation stage, in particular detailed routing. However, most existing studies for lithography-friendly routing suffer from either huge run-time due to the intensive lithographic computations involved, or severe loss of quality of results because of the inaccurate predictive models. In this paper, we propose *AENEID* - a fast, generic and high performance lithography-friendly detailed router for enhanced manufacturability. *AENEID* combines novel hotspot detection and routing path prediction techniques through modern *data learning* methods and applies them at the detailed routing stage to drive high fidelity lithography-friendly routing. Compared with existing litho-friendly routing works, *AENEID* demonstrates 26% to 66% (avg. 50%) of lithography hotspot reduction at the cost of only 18%-38% (avg. 30%) of run-time overhead.

Categories and Subject Descriptors

B.7.2 [**Hardware, Integrated Circuit**]: Design Aids

General Terms

Algorithms, Design and Performance

Keywords

Detailed Routing, Design for Manufacturability, Hotpost Detection, Data Learning

1. INTRODUCTION

As a consequence of the shrinking technology process and the increasing chip complexity, the design and manufacturing cycles start to see more and more interactions for modern VLSI ICs. As lithography-induced yield is a critical factor to optimize for volume manufacturing of nanometer ICs, various resolution enhancement techniques (RET), such as optical proximity correction (OPC), off-axis illumination (OAI), phase shift mask (PSM), source-mask optimization (SMO), and double patterning technology (DPT) are employed to enhance layout printability and yield.

However, RET during mask synthesis alone is not enough due to widening manufacturing gaps [1], which require increasing cooperation of physical design methods to generate lithography-friendly layouts to start with. Several works have been proposed to incorporate accurate lithographic models or predictive models into physical design stages, in particular the routing stages, to ensure layout printability. In [2], post-routing ripup-&-reroute was proposed to remove lithography hotspots, guided by fast lithography simulations. In [3], an OPC-cost aware router was proposed utilizing post-layout OPC models characterized by quasi-inverse lithography techniques. OPC-aware maze routing methods are also proposed based on multi-constrained shortest path optimization with sub-gradient method [4] and optical proximity error (OPE) metrics [5]. In [6], a litho-friendly detailed router was proposed based on weak grid types of predictive OPC metrics that are updated on a per-grid basis. However, these existing studies all suffer from one or more of the following issues: (1) huge run-time due to accurate but slow lithographic simulations; (2) over simplified predictive models severely limit the solution space; (3) particularly designed to work only for certain classes of RETs (e.g., OPC, etc.) under certain data fitting assumptions, but not generic enough to handle new types of RETs (e.g., Litho-Etch-Litho-Etch Double Patterning, Self-Aligned Double Patterning, Sub-Resolution Assist Features Insertion) and evolving manufacturing conditions.

To address these concerns, modern graph theory and data mining/learning methods have recently been adopted to build reliable and high performance lithography hotspot detection engines. [7] proposed a graph pattern based hotspot filtering method to reduce the hotspot candidate searching space without compromising the overall detection quality. Concept of range pattern is later introduced in [8] to accurately and compactly represent process hotspots. In [9], ripup-&-reroute techniques are proposed for hotspot removal, utilizing a predefined pattern matching library. Although generally fast, issues with graph/pattern matching techniques lie in detection coverage and scalability: (1) hotspot patterns are very difficult to enumerate - too much patterns may constrain the solution space, while too few patterns suffer from high detection false-alarms; (2) high false-alarms in the physical design stages introduce heavy workload for post synthesis correction; (3) pattern definition is highly dependent on design rules and process technology, therefore increases the development burden as technology evolves.

With data mining techniques, further improvements were later made in [10], where a support vector machine (SVM)-based hotspot detection method is utilized through performing 2D distance transform and histogram extraction. Also in [11], SVM is employed for hotspot detection through classification of layout density metrics. However, the issues with the above approaches lie in run time and detection coverage, since 2D transforms and density extractions can be expensive to perform, while detection windows for the layout images are hard to anchor for full chip level detections. In [12], critical hotspot signature is proposed and extracted through certain special edge-based metrics. Although such edge-based extractions operate much faster compared with [10,11], its chip-level prediction still faces similar issues, such as scanning window

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DAC 2011, June 5–10, 2011, San Diego, California, USA.
Copyright 2011 ACM 978-1-4503-0636-2/11/06 ...$10.00.

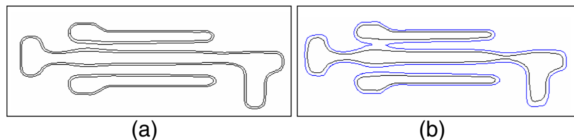


Figure 1: A case of RET dependent layout printability

coverage, etc. To further improve runtime and detection coverage, a hierarchically refined machine learning framework is proposed in [13] for fast speed high performance hotspot detection using both Artificial Neural Network (ANN) and SVM classifications.

Yet there has been no work so far to apply such data learning hotspot detection framework directly to physical design in a correct-by-construction manner. This is mainly because hotspot detection requires post layouts as inputs, thus the lithography cost cannot be updated in time to guide the physical design. To address the aforementioned limitations, in this paper for the first time we propose *AENEID*, a lithography-friendly detailed router that is seamlessly integrated with modern data learning techniques and novel predictive models for litho-aware path prediction.

The rest of the paper is organized as follows, in Section 2, we elaborate the motivation of this work and summarize our main contributions. In Section 3 we explain the formulation and overall flow of *AENEID*. In Section 4, we describe the key novel techniques employed for the hotspot detection and routing path prediction procedures. In Section 5 we evaluate *AENEID* with various industry strength benchmarks under industry strength RET, we also present and discuss the experimental results compared with some existing study. We conclude the paper in Section 6.

2. MOTIVATION AND CONTRIBUTIONS

In this section, we explain the post-RET hotspot detection dilemma in the detailed routing stage and introduce two most critical considerations in *AENEID*: (1) fast and accurate lithographic hotspot characterization that drives the litho-friendly router; (2) look-ahead routing path prediction that adjusts the litho-cost assigned to the router for enhanced yield and routability.

With the rapid advancement of modern lithography technology and RET techniques, real process hotspots are getting less as process matures and RET improves, however real/residual lithography hotspots are becoming more critical to the nanometer IC design. In the mask synthesis stage, process hotspots are highly dependent on the RETs employed: an effective RET can print a layout pattern reliably (Fig. 1(a)), whereas a poorly setup RET may generate many false-alarm "hotspots" (Fig. 1(b)) that will not be present under real industry-strength manufacturing condition [13]. With such considerations, data learning models [10–13] become especially suitable for guiding detailed routers due to their satisfactory performance. However, they must be properly sped up before incorporating into inner design loops, due to the strict run-time requirement in the routing stage.

In addition to strict run-time requirement, another challenge lies in the proper modification of data learning kernels to comply with the incremental mechanism in the detailed routing stage. With the example in Fig. 2(a)-(b), we show a hotspot detection dilemma in the detailed routing stage.

In Fig. 2(a), the shadowed regions are metal blockages; Pin1-Pin2, Pin3-Pin4 are 2 nets yet to be routed in the detailed routing stage, which means at the current step the bottom-right corner is a blank region. On one hand, the detailed routing paths from Pin1 to Pin2 and from Pin3 to Pin4

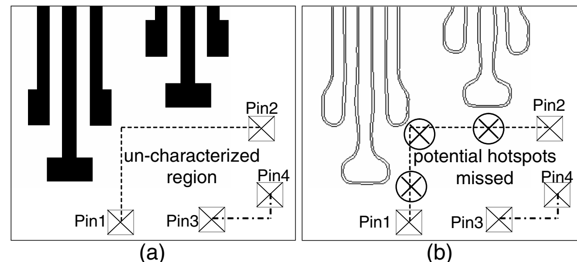


Figure 2: The lithography hotspot detection dilemma in the detailed routing stage

are to be optimized depending on the cost updates to be provided by a hotspot detection engine; on the other hand, a hotspot detection engine must first have a routing path in order to provide the routing cost updates. In other words, the lithography cost cannot be updated in time to guide routing in a correct-by-construction manner. Consequently, we are left with a un-characterized region shown in Fig. 2(a). Within this region, detailed routing might be performed in a lithography unaware manner which will potentially result in large amount of lithography hotspots (Fig. 2(b)).

To eliminate un-characterized regions, we propose for the first time a novel set of predictive formulae on top of existing hotspot detection kernels, to predict the routing path with the least *expected* lithographic cost. Such a set of formulae is developed *a priori* through accurate lithographic simulations of various layout samples at a one time cost. Once completed, it can be rapidly applied to compensate existing hotspot detectors in the detailed routing stage. More details will be explained in Section 4.

With these motivations and considerations, we propose *AENEID*, the first correct-by-construction detailed router formulation incorporating advanced data learning techniques and routing path predictive formulae for lithographic yield improvement. We summarize the main contributions of *AENEID* as follows,

- We propose a correct-by-construction detailed routing flow that is *generic* and *adaptive* to any existing RETs without any lithographic simulations involved during the routing stage.
- We employ modern data learning techniques for fast and accurate lithography hotspot detection.
- We develop lithography-friendly routing path prediction model to resolve the hotspot detection dilemma in the detailed routing stage.
- We integrate the hotspot detection and routing path prediction techniques into a scalable and high performance litho-friendly detailed router and achieve very promising results.

3. AENEID FORMULATION & OVERALL FLOW

3.1 Problem Formulation

The objective in our detailed routing is to minimize total wirelength (number of grids in set P of total routing paths) with the constraint to keep the lithography cost $litho(e)$ on each routing grid e under a given threshold L. Therefore we can formulate our lithograph-friendly detailed routing problem as follows:

$$\begin{aligned} \min_{P}: &\quad \textstyle\sum_{e \in P} 1 \\ s.t: &\quad litho(e) \le L \quad \forall e \in P \end{aligned} \tag{1}$$

If we treat the costs for all the grids as a weight-vector, this problem can be viewed as a multi-constraint shortest path (MCSP) problem [14] which is proven to be NP-hard. Lagrangian relaxation can be used to solve MCSP by relaxing the constraints into the objective function by introducing Lagrangian multiplier λ_e as the weights on the constraints. Then we can relax the original formulation as [4, 6]:

$$\min_P \left\{ \sum_{e \in P} 1 : litho(e) \leq L, \forall e \in P \right\}$$
$$\geq \max_\lambda \min_P \sum_{e \in P} 1 + \lambda_e(litho(e) - L) : \lambda_e \geq 0 \qquad (2)$$

Equation(2) shows that the optimal solution of Equation(1) can be obtained by maximizing the lower bound of the following Lagrangian subproblem:

$$\min_P : \quad \sum_{e \in P} 1 + \lambda_e(litho(e) - L) \qquad (3)$$
$$s.t. : \qquad \lambda_e \geq 0 \qquad \forall e \in P$$

After assigning $1 + \lambda_e litho(e)$ as the weight of each grid e, Equation(3) can be solved by min-cost path algorithm. Then we can iteratively solve Equation(3) and adjust the Lagrangian multiplier to obtain the maximum lower bound for the relaxed problem in Equation(2). Based on the formulation here, we explain our detailed router in Section 3.2.

3.2 Overall Flow

Since maximizing Equation(3) is a convex programming problem, we can apply subgradient method to solve it. As shown in Fig. 3, the key steps of *AENEID* are as follows.

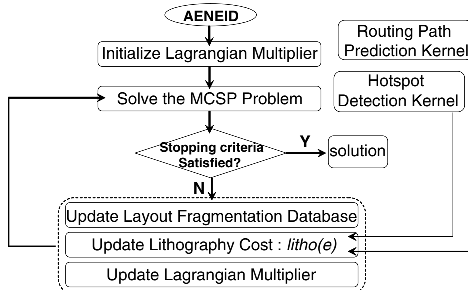


Figure 3: ***AENEID*** **detailed routing flow chart**

- **Step 1**: Initialize Lagrangian multipliers λ_e with small non-zero values for faster convergence.
- **Step 2**: For each net, solve Equation(3) by finding its min-cost path. A*-tree technique is used here to prevent unnecessary path finding.
- **Step 3**: update λ_e by $\max(0, \lambda_e + \theta \times litho(e))$ according to the result from **Step 2**. Here the update of $litho(e)$ consisting of two parts, whose values are calculated by 2 pre-established kernels.
- **Step 4**: repeat from **Step 2** until the maximum iteration number is reached. Finally we can obtain a convergent solution.

In Fig. 3, **Step 3** is the most critical step for *AENEID*, since it is in charge of litho-cost updates for every loop of the detailed routing. Inside the *Lithography Cost Update* procedure of **Step 3** are the two most important contributions of this work, namely the *Hotspot Detection (HD)* technique and the *Routing Path Prediction (RPP)* technique, which we will elaborate in the following section.

4. DATA LEARNING AND HOTSPOT PREDICTION

In the detailed routing flow in Section 3.2 we represent the layout contents (polygons) with fragmentation-structured database meanwhile maintain a litho-cost map for the entire set of routing grids according to the current routing density. The lithography cost $litho(e)$ is then iteratively updated for each grid e. In *AENEID*, $litho(e)$ is calculated in two parts in Equation(4):

$$litho(e) = litho(e)^{HD} + litho(e)^{RPP} \qquad (4)$$

where $litho(e)^{HD}$ is calculated using the *Hotspot Detection* technique and $litho(e)^{RPP}$ is calculated with the *Routing Path Prediction* technique. Depending on whether the neighborhood has been populated with wire segments or not, $litho(e)^{HD}$ and $litho(e)^{RPP}$ combine hotspot litho-cost of the current step layout and of the future routing steps together to provide enhanced lithography-friendliness and improved routability. Unlike the existing works such as [6] that only considers $litho(e)^{HD}$ term when updating litho-cost metrics, *AENEID* benefits greatly from such a combination of litho-costs that we propose, as we will show later in Section 5.

4.1 Hotspot Detection Technique

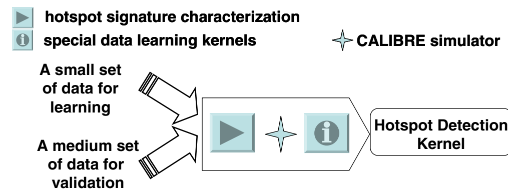


Figure 4: **Development of the *HD* Kernel**

Hotspot Detection technique aims at establishing a compact kernel model to calculate the degree of printability (manufacturability) of certain layout pattern via data mining and classification methods. As shown in Fig. 4, we develop our *HD* kernel based on [13], with the following modifications: (1) fine-tuned parameters for enhanced speed and detection accuracy of Support Vector Machine classifier; (2) adjustment of the "hotspot signature metrics" definitions to better take care of line-end and jog characterizations; (3) instead of using an effective radius r, we query the top N nearest neighbor fragments from the database and characterize the context accordingly for better focused meanwhile faster hotspot detection. Please refer to Section 5 for more details regarding the classification and validation accuracies. Please refer to [13] for details of Fig. 5 regarding layout fragmentation.

Once established, the *HD* kernel will be used in the detailed routing stage for calculating $litho(e)^{HD}$ which is the 1st part

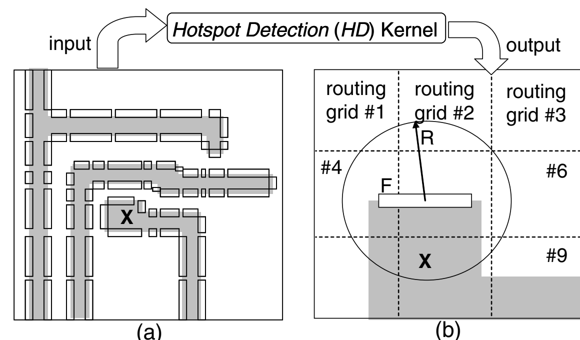


Figure 5: **Applying the *HD* Kernel for litho-cost update**

of $litho(e)$. As depicted in Fig. 5(a)(b), *HD* kernel intakes the fragmentation database and returns a quick estimation of layout printability by characterizing the context of fragment F. Since the *HD* kernel is derived *a priori* at a one time cost, it can be used as a quick look-up knowledge base to characterize hotspot conditions even in the inner design loops.

Since *HD* kernel calculates litho-cost based on database fragments, its result $litho(F)$ must be properly assigned to $litho(e)$ to guide the grid-based detailed routing. Here we use the method shown in Fig. 5(b) to map $litho(F)$ onto routing grids. Given an effective radius R, $litho(F)$ is used to update all the unrouted grids that lie within the radius, i.e., grid 1, 2 and 6. In section 4.3 we will show more details on the fragmentation database update and grid-based cost assignment.

As explained in Section 2, *HD* itself is not sufficient to guide a correct-by-construction router since it only characterizes the contexts of already existing polygons in the layout. We need the *Routing Path Prediction (RPP)* technique to further enhance the lithography-printability on the routes that fall into un-characterized regions.

4.2 Routing Path Prediction Technique

RPP technique is very important to resolve un-characterized regions and further improve the router's lithography friendliness. In the follow subsection, we describe in detail the development of *RPP* kernel and its application to calculate $litho(e)^{RPP}$ which is the 2nd part of lithography cost $litho(e)$.

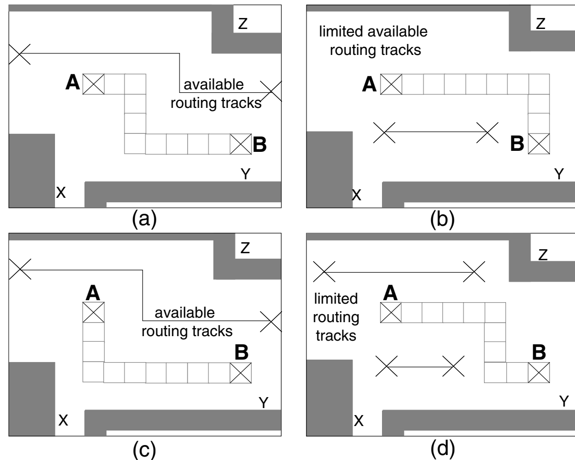


Figure 6: A motivational example for lithography-friendly routing path prediction

The development of *RPP* involves intensive lithographic computations since it is in nature a greedy searching algorithm. For this paper, we pre-establish *RPP* at a one time cost and build a multi-objective compact model on top of its knowledge base to allow fast applications inside *AENEID*'s inner design loop.

RPP's optimization objective is given as follows:

$$min\{E[\sum_{j=i+1}^{N} litho(route^j)|route^i]\}, w.r.t \quad i \qquad (5)$$

where $route^i$ is the current iteration in the detailed router at which point all the routes from 0 to $i-1$ are already fixed. $E[\cdot]$ denotes the mathematical expectation operation. Equation(5) aims to find the best route i to take among possible routes i to N, so that the overall potential lithography printability is maximized with possible subsequent routes taken into considerations.

With the example shown in Fig. 6(a)-(d), we elaborate the development of *RPP*. Given certain characterized layout context (blockages X,Y,Z) and a pair of pins to route (A,B), we illustrate several possible routes in Fig. 6(a)-(d). With each possible route, the number of remaining available routing tracks also varies. In Fig. 6(a)(c), there are still available tracks running from left to right side, but for Fig. 6(b)(d), all the tracks are limited to local scale. In this case, *RPP* will be established in 3 steps. **Step 1**: explore a wide range of possible routes given the available routing resources. Virtual blockages are placed to help generate a variety of alternative routing paths for Equation.5; **Step 2**: run accurate lithographic simulations for all (a)-(d) layout patterns and assess printability; **Step 3**: update two priority queues based on **Step 2** and recommend/encourage a preferable route that: (1) gives the least number of hotspots; (2) provides the most number of available tracks so that subsequent routes can be made easier. Tradeoffs need to be sought if the two queues return different results.

Due to the huge data volume of the resulting knowledge base, we employ a robust neural network classifier to construct the actual prediction model to incorporate into *AENEID*. With the *RPP* kernel ready, we apply it to predict routing paths given the context environment of a net to be routed. In the detailed routing engine, $litho(e)^{RPP}$ lying on the paths that are favored by *RPP* kernel will be adjusted to encourage litho-friendly routing. Thus the $litho(e)^{RPP}$ is updated iteratively inside the router.

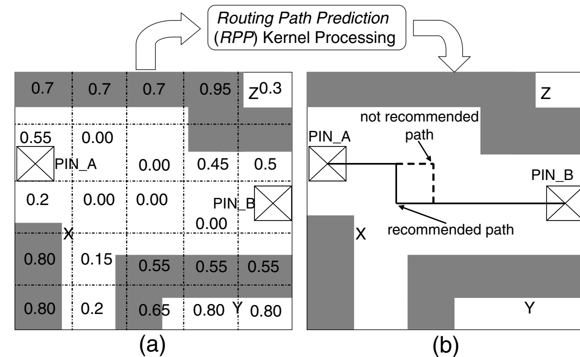


Figure 7: The fragment-based litho-cost map update based on *Path Prediction (RPP)*

In Fig. 7, we show our layout density based multi-objective neural network model. In Fig. 7(a), the input to *RPP* kernel is a vector consists of the pin locations of the net to be routed and the density grid (not routing grid) array whose elements signify blockage densities. *RPP* returns a set of grid locations (region) of preferred routes under lithography-friendly considerations. Based on this information, all the $litho(e)$ touching this path or region are updated accordingly. Please refer to Section 5 for further details regarding kernel training and validations.

4.3 Fragmentation-based Update

We build our layout database using fragment data structures due to the unique advantages of fragmentation based layout pattern characterization. In this subsection, we describe our layout representation techniques in detail.

Conceptually, fragments are defined as vectors lying on the edges of all polygons in the layout. A fragment based layout database makes it very efficient to query entries such as nearest neighbors and polygon width (max distance between internally facing fragments), etc. It also provides satisfactory

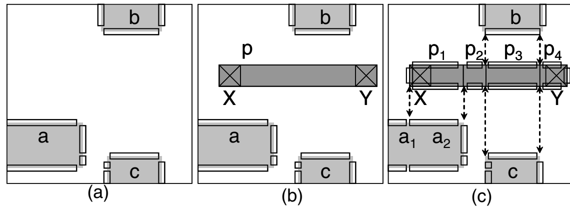


Figure 8: Illustration for the fragment database update

analyzing resolution (polygon edges) and detection coverage (the whole layout) [13]. Under such a database structure, we use two key techniques for fast data access speed, namely, the *sweep line* algorithm and the *Red-Black Binary Search Tree.*

AENEID requires each fragment to keep updating its neighboring fragments information in real time, since new fragments are introduced into the database as each additional net is routed. Obviously this updating procedure would be invoked frequently whenever the status (occupied/non-occupied) on the routing grid e is changed, thus it needs to be properly taken care of to minimize the runtime degradation. Under such a consideration, we propose a *sweep line* algorithm to obtain the neighboring fragments for a given routing path.

Take Fig. 8 as an example, assume there are three existing fragments a, b and c on the routing grid shown in Fig. 8(a). The arrow appears if two fragments are neighboring to each other. If a new route p is added as shown in Fig. 8(b), all fragments affected by p need to be updated. The algorithm sweeps the routing grid from left to right, each time a new fragment is detected by the sweep line, it checks if there is a pre-swept fragment neighboring to it and do the updating if necessary. Fig. 8(c) shows the final result and we can also see that p and fragment a are decomposed to accurately reflect the neighboring situation. By using this algorithm, the updating can be done in $O(\log N)$, where N is the total fragment number in the database. Overall, the effort for the router to interact with our litho-cost related models is $O(k \log N)$ time, where k is the total number of rip-up and route operations.

To effectively update the litho-cost map and identify/query fragments by certain specified layout region, we use an RB-tree to store the information of all fragments. By the property of RB-tree, locating fragments within a given region can be done in $O(\log N)$ time, where N is the number of total fragments in the routing grid. Therefore we can update the litho-cost map in $O(\log N)$ time whenever a path is routed or ripped up.

5. EXPERIMENTAL RESULTS

We implemented *AENEID* in C++ and evaluated it with testing cases in 45nm M1-M2 technology process under industry strength Optical Proximity Correction recipes. All simulations are performed on Linux workstations with 4GB memory and Intel Xeon 2.66GHz CPU. In the layout validation baselines, we employed accurate lithographic simulations to locate all the real hotspots based on an edge placement error (EPE) threshold of 8nm. Inside *AENEID*, we used an aggressive lithography cost upper-bound L to yield the least number of hotspots. *AENEID* can be further sped up by fine-tuning L properly.

5.1 Kernel Training & Validation

Accordingly to the flow shown in Fig. 9, we implement the data learning techniques in C++ and complete the knowledge base update via iterations of training and validation.

For the *HD* kernel model, we use a ν-Class Support Vector Machine classifier. We perform supervised data learning over an 80X80 um^2 design consisting of 40K sample patterns

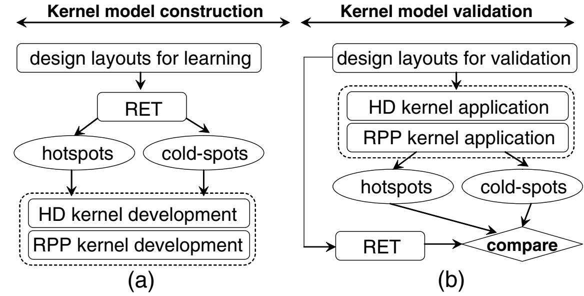


Figure 9: Kernel training and validation procedures

properly labeled by accurate lithographic simulators. Then the established *HD* kernel is validated and tested with various different design samples under the same process technology. *HD* shows 92% of accuracy and 3% of false-alarms for the 40K training patterns, about 88% of accuracy and 8% of false-alarms among another 70K new testing patterns. Such performance is very satisfactory compared with existing studies. The total run-time for the *HD* kernel establishment has a one time cost of around 15 min.

For the *RPP* kernel, we first use the proposed greedy search algorithm combined with accurate lithographic simulations. Then we build a multi-objective neural network data learning model on top of the derived database using 3 hidden layer neurons and a resilient conjugal gradient learning function with MSE target 0.1. The greedy search is carried out over an 80X80 um^2 design. The neural network model is trained and validated on 200K sample vectors, tested on another 100K samples. *RPP* demonstrates an average 87% accuracy in the training set, and 80% of accuracy in the testing set. Considering the fact that density grids are usually set much larger than the routing grids, *RPP* performs well within its error tolerance. The total run-time for the *RPP* kernel establishment has a one time cost of around 3 hours.

5.2 AENEID Experimental Results

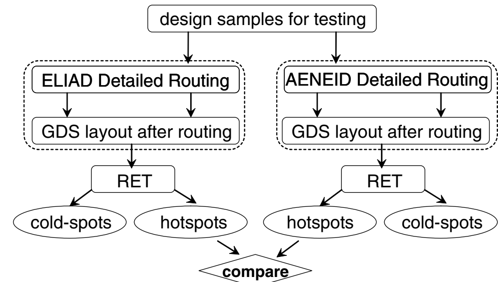


Figure 10: The validation flow for ELIAD and *AENEID*

We test *AENEID* and compare it with [6] according to the flow chart shown in Fig. 10, where accurate lithography simulations are employed for hotspots calibration based on 8nm of EPE threshold under industry strength OPC setups with 45nm M1-M2 process.

Testing Benchmarks of AENEID: Table 1 lists 3 industry-strength benchmarks employed to evaluate *AENEID*. These testing cases have not gone through the training or validation

Table 1: Circuit benchmarks for testing *AENEID*

Benchmarks	CK1	CK2	CK3
Layout Size	50X50um^2	100X100um^2	160X160um^2
Nets to route	0.45K	1.48K	3.4K
M1 Blockage #	1K	8.8K	13.1K
M1 Fragment #	12.2K	41K	152.6K
M2 Blockage #	0.14K	0.47K	2K
M2 Fragment #	0.56K	1.9K	8.3K

Table 2: Result comparison between *ELIAD* [6] and our proposed *AENEID*

	ELIAD						AENEID											
							HD						HD + RPP					
Circuit	CK1		CK2		CK3		CK1		CK2		CK3		CK1		CK2		CK3	
Circuit size um^2	50^2		100^2		160^2		50^2		100^2		160^2		50^2		100^2		160^2	
Wire-length um	859.8		5509.0		24789.2		859.3		5502.0		24797.0		859.1		5502.0		24797.5	
Run-time sec	6		297		2773		8		409		3291		8		400		3279	
Run-time overhead %	-		-		-		33		38		19		33		35		18	
Metal layer	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2
Hotspot #	17	3	65	10	162	23	11	2	34	7	90	17	8	2	22	5	58	15
Hotspot reduc %	-	-	-	-	-	-	35	33	48	30	44	26	53	33	66	50	64	35
Avg. hotspot reduc %	-						36						50					
Avg. extra run-time %	-						30						29					

process of the *HD/RPP* kernels thus are considered generic and unbiased. Also in Table 1 we show the numbers of initial routing blockages and fragments on M1 and M2 layers, which come from the placement of standard cells.

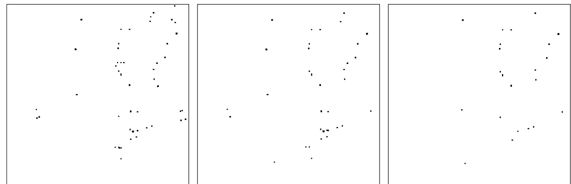
In Table 2 we list the experimental result comparisons between ELIAD [6] and *AENEID*, in terms of hotspot number reduction, total wirelength and run-time. On *AENEID*s side, we run the simulation with 2 options in the $litho(e)$ update step: (1) *HD* only; (2) *HD + RPP*.

There are several key observations to make in Table 2. First, compared with ELIAD, *AENEID* ***HD*** demonstrates about 26%-48% (avg. 36%) hotspot reduction at only 30% of average extra run-time; while *AENEID* ***HD+RPP*** shows 35%-66% (avg. 50%) hotspot reduction at only 29% of average run-time overhead. This shows us: (1) **HD** kernel proves to be compact and accurate than the predictive model used in ELIAD; (2) **RPP** kernel resolves the un-characterized regions thus further reduces the hotpots and improves printability.

Second, **HD+RPP** results in even smaller run-time overhead than **HD**. This is mainly because the **RPP** kernel has reduced the number of rip-up and re-route nets, thus ends up saving some run-time than **HD** alone.

Third, *AENEID* demonstrates similar wirelength to ELIAD, this is mostly because we employ a similar optimization formulation. This also shows us that **HD** and **RPP** kernels did not bring obvious degradation to the total wirelength.

In Fig. 11, we show the hotspot calibration result visually with: (a) ELIAD on CK3; (b) *AENEID HD* on CK3; and (c) *AENEID HD+RPP* on CK3. Combined with Table 2, we conclude that *AENEID* demonstrates greatly enhanced layout printability at acceptable run-time overhead, meanwhile its flow is generic and adaptive to RETs (not only limited to OPC/ORC, etc.). For more break-down details of the simulation, please refer to Table 2.



(a) ELIAD on CK3 (b) AENEID-HD on CK3 (c) AENEID-HD+RPP on CK3

Figure 11: Comparisons of lithography hotspot numbers between ELIAD and AENEID on CK3

6. CONCLUSION

In this paper for the first time, we proposed *AENEID* - a fast, generic and high performance lithography-friendly detailed router for enhanced manufacturability at advanced process technology nodes, offering advantageous features as follows, (1) it combines modern data learning methods and novel hotspot prediction techniques to develop compact kernel models through analyzing and learning from a relatively small set of lithography hotspot samples under real industry strength manufacturing conditions; (2) it applies the pre-established kernels at the detailed routing stage to drive fast high fidelity lithography-friendly interconnect synthesis; (3) its flow and data learning procedures are generic to any RETs, not just limited to certain design patterns or OPC setups. *AENEID* is simulated and compared with existing state-of-the-art studies over various industry strength testing cases, demonstrating a significant 22%-66% (50% on average) of lithography hotspot reduction at the cost of only 18%-38% (30% on average) of run-time overhead.

7. ACKNOWLEDGMENT

This work is supported in part by SRC, NSF Career Award and equipment donations from Intel.

8. REFERENCES

[1] Ed. Roseboom, M. Rossman, F.-C. Chang, and P. Hurat. Automated Full-Chip Hotspot Detection and Removal Flow for Interconnect Layers of Cell-Based Designs. In *Proc. of SPIE*, volume 6521, 2007.

[2] J. Mitra, P. Yu, and D. Z. Pan. RADAR: RET-aware detailed routing using fast lithography simulation. In *Proc. Design Automation Conf.*, 2005.

[3] T.-C. Chen, G.-W. Liao, and Y.-W. Chang. Predictive Formulae for OPC with Applications to Lithography-Friendly Routing. In *Proc. Design Automation Conf.*, 2008.

[4] L.-D. Huang and M. D. F. Wong. Optical Proximity Correction (OPC). In *Proc. Design Automation Conf.*, 2004.

[5] Y.-R. Wu, M.-C. Tsai, and T.-C. Wang. Maze Routing with OPC Consideration. In *Proc. Asia and South Pacific Design Automation Conf.*, 2005.

[6] M. Cho, K. Yuan, Y. Ban, and D. Z. Pan. ELIAD: Efficient Lithography Aware Detailed Routing Algorithm with Compact and Macro Post-OPC Printability Prediction. In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 09.

[7] A. B. Kahng, C.-H. Park, and X. Xu. Fast Dual Graph based Hotspot Detection. In *Proc. of SPIE*, volume 6349, 2006.

[8] H. Yao, S. Sinha, C. C. Chiang, X. Hong, and Y. Cai. Efficient Process Hotspot Detection using Rang Pattern Matching. In *Proc. Int. Conf. on Computer Aided Design*, 2006.

[9] F. Yang, Y. Cai, Q. Zhou, and J. Hu. SAT Based Multi-Net Rip-up-and-Reroute for Manufacturing Hotspot Removal. In *Proc. Design, Automation and Test in Eurpoe*, 2010.

[10] D. Gagi Drmanac, F. Liu, and L.-C. Wang. Predicting Variability in Nanoscale Lithography Processes. In *Proc. Design Automation Conf.*, San Francisco, CA, 2009.

[11] J.-Y. Wuu, F. G. Pikus, J. A. Torres, and M. Marek-Sadowska. Detecting Context Sensitive Hot Spots in Standard Cell Libraries. In *Proc. of SPIE*, 2009.

[12] D. Ding, X. Wu, J. Ghosh, and D. Z. Pan. Machine Learning based Lithographic Hotspot Detection with Critical Feature Extraction and Classification. In *International Conference for Integrated Circuit Design Technology*, 2009.

[13] D. Ding, J. A. Torres, F. G. Pikus, and D. Z. Pan. High Performance Lithographic Hotspot Detection using Hierarchically Refined Machine Learning. In *Proc. Asia and South Pacific Design Automation Conf.*, 2011.

[14] J. Dong, J. Zhang, and Z. Chen. Neural Network based Algorithm for Multi-Constrained Shortest Path Problem. In *Proc. Int. Symp. on Neural Networks*, 2007.