The Nebulous Hotspot and Algorithm Variability

Alfred K. K. Wong[a] and Edmund Y. Lam[b]

[a]Magma Design Automation, Inc., 1650 Technology Drive, San Jose, CA 95110, U. S. A.*
[b]Department of EEE, The University of Hong Kong, Pokfulam Road, Hong Kong†

ABSTRACT

Computation lithography relies on algorithms. These algorithms exhibit variability that can be as much as 5% ($1\,\sigma$) of the critical dimension for the 65-nm technology. Using hotspot analysis and fixing as an example, such variability can be addressed on the algorithm level via controlling and eliminating its root causes, and on the application level by setting specifications that are commensurate with both the limitations of the algorithms and the goals of the application.

Keywords: algorithm variability, computation lithography, LPC, hotspot

1. INTRODUCTION

1.1 Lithography process check and design rule check

Lithography hotspots, those that are found through lithography process check (LPC),[1,2] and design rule violations[3] are cousins, sharing some common characteristics while differing in other aspects. Both relate to the robustness of physical designs; existence of hotspots or violations indicates propensity to yield loss. Despite yield being an inherently continuous quantity, for design convenience and expediency, hotspots and design rule violations have well-defined pass-fail boundaries. A sharp demarcation is (artificially) placed between violations and non-violations, and between hotspots and non-hotspots.

Given a layout with a design rule error, it is usually straightforward to understand the cause of the violation. Such cannot be said for hotspots. Except for the simplest of cases where guidelines and rules-of-thumb are applicable, subtleties of optical imaging and pattern transfer preclude use of intuition and elementary logic. In fact, complexity of low-k_1 lithography is the original motivation for model-based verification.

The non-intuitive nature of LPC should not hamper its utility; numerical computation is increasingly complementing instinctive reasoning in sub-100-nm integrated circuit creation. Model accuracy is perhaps a cause of greater concern. State of the art lithography models still fall short in critical dimension prediction of two-dimensional patterns and of non-nominal exposure conditions.[4–6] Resist collapse is also difficult to simulate properly.[1] Nevertheless, design rule check (DRC) is not immune to inaccuracies: design rules and their parameters are also defined with uncertainty and incertitude.

What truly distinguishes LPC from DRC is the former's imprecision. This is at present one of the two greatest hindrances to wholesale adoption of LPC in the design flow.‡ Imprecision in the present context means fluctuation of simulation results caused by LPC algorithm imperfections or mismatches. Two identical layout configurations may have inconsistent hotspot classification, with one indicating a hotspot while the other does not. Sources of variability may include process model asymmetry, image interpolation, target layer deviation due, for example, to hierarchical operations, OPC iteration, and contour computation. Cutting-edge DRC algorithms do not exhibit such impreciseness.

*The work described in this paper was partially performed while the author was associated with Magma Design Automation, Inc.

†The work described in this paper was partially supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Numbers 713906 and 713408).

‡The other obstacle is the general lack of evidence for or conviction in the necessity of LPC.

Design for Manufacturability through Design-Process Integration III, edited by Vivek K. Singh, Michael L. Rieger,
Proc. of SPIE Vol. 7275, 727509 · © 2009 SPIE · CCC code: 0277-786X/09/$18 · doi: 10.1117/12.816449

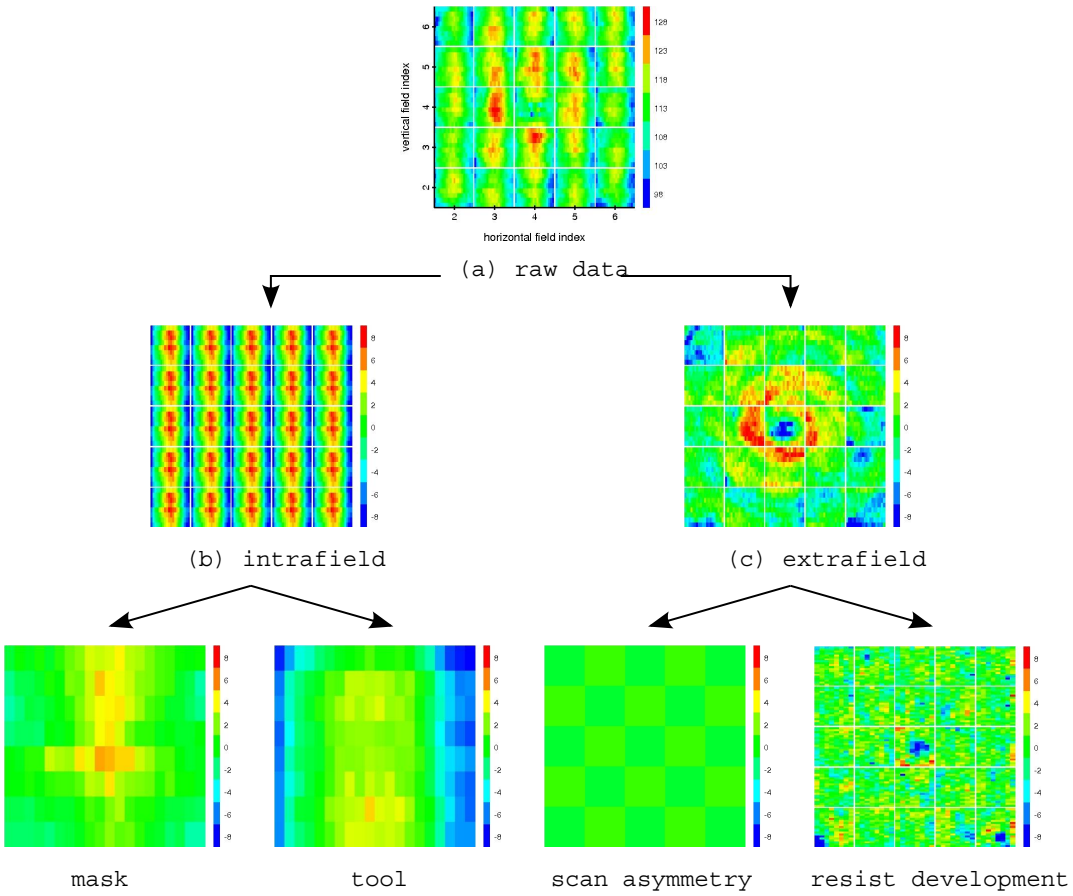


Figure 1. Measured across-wafer critical dimension variation.[7]

1.2 Analogy between algorithmic and manufacturing variability

The nebulous hotspot is a manifestation of the more general issue of variability in computation lithography algorithms. Before embarking on our examination, it is instructive to refer to Fig. 1(a), which shows the measured across-wafer critical dimensions of nominally identical isolated lines. The color shade at each point represents the electrical linewidth of the isolated line at that wafer location. Through a series of analysis, the original distribution [Fig. 1(a)] is decomposed into intrafield [Fig. 1(b)] and extrafield [Fig. 1(c)] components, which are then further partitioned into photomask errors, scanning errors of exposure systems, aberrations, and development non-uniformity.[7] As a result, root causes of linewidth fluctuation can be quantified separately and remedied individually, thereby reducing the overall variability.

Such analysis and quantification also necessitates and enables the setting of proper design margins, in order to ensure high-yielding and robust circuits. The design margin, or, indirectly, achievable circuit performance, is intimately tied to the amount of linewidth variation. Higher degrees of variability require larger margins, resulting in lower performance, and vice versa. Circuits must be designed with margins commensurate with the level of variability. Forcing circuit performance without proper regard for process fluctuation is a path to a miserable end.

The relationship between computation lithography algorithms, such as image calculation, and applications, such as lithography design-for-manufacturability (DFM), is analogous to that between fabrication and circuit design: algorithms (semiconductor processing) is the foundation on which applications (circuits) are built; and applications (circuit design) should be cognizant of algorithmic (manufacturing) variability. On the one hand, we need to analyze and decompose sources of variation in order to quantify, monitor, and minimize such fluctuation.

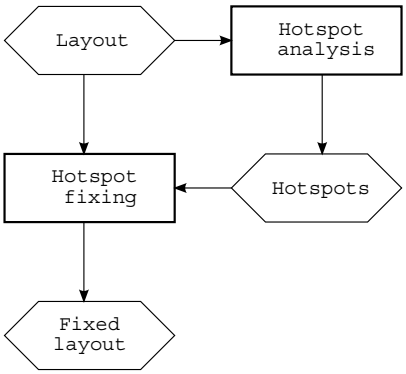


Figure 2. Lithography physical DFM comprises hotspot analysis and hotspot fixing.

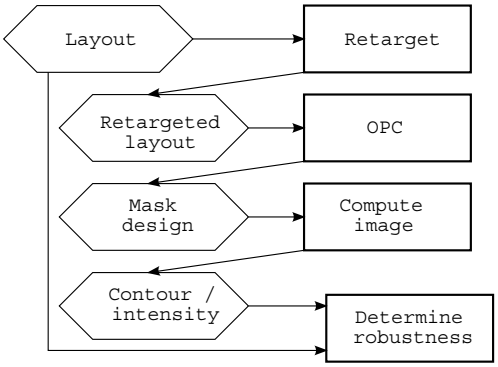


Figure 3. A typical LPC flow.

On the other hand, given a particular level of variability, we can devise reasonable specifications that are at once realistic, practical, and useful, without imposing unrealistic requirements or delusive expectations.

1.3 Lithography physical DFM

As a vehicle for investigation and exposition, let us consider lithography physical DFM, a process, illustrated in Fig. 2, that includes hotspot analysis (LPC)—identifying regions of a layout that have relatively poor lithography latitude—and hotspot fixing—adjusting the layout to improve process robustness of the identified regions. For our present purposes, we are concerned with algorithmic variability during hotspot analysis and, given such fluctuation, sensible measures we can adopt for hotspot fixing.

2. LPC VARIABILITY

It is worthwhile, prior to examining sources of LPC variability, to delineate the concept of *constancy*. LPC behavior falling outside the bounds of constancy is then variability.

We expect that, within the confines of lithography modeling, identical features with identical layout environments would have identical lithography characteristics. For practical purposes, identical surroundings means that the neighborhood layouts are identical at least up to a distance on the order of the optical interaction range[8]

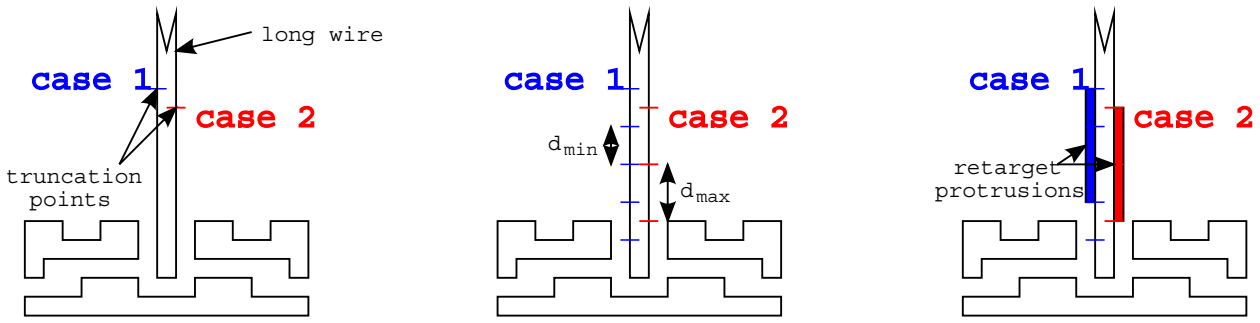
$$r_{\text{optical\_interaction}} \sim \frac{1}{\sigma} \times \frac{\lambda_0}{\text{NA}},$$

where λ_0, NA, and σ respectively denote the wavelength in vacuum, the numerical aperture, and the partial coherence factor of the exposure system. For a 193-nm system with a numerical aperture of 1.2, two features should have the same hotspot categorization if their environments within

$$\mathbf{O}(10) \times \frac{1}{0.8} \times \frac{193}{1.2}\,\text{nm} \approxeq 2,000\,\text{nm}$$

are identical.

Such expectation is not always met in practice. In a typical LPC flow shown in Fig. 3, the original layout first undergoes retargeting that may include global and selective sizing. Optical proximity correction (OPC) is then applied on the retargeted layout, producing a photomask design; images of the photomask are computed using a model of the lithography process. From the images are derived contours, critical dimensions, and process latitude information, based on which hotspot decisions are made. Any of these steps may be a source of variability.



(a) Layout with long wire. (b) Segmentation differences. (c) Retargeting differences.

Figure 4. Measures for improving computation throughput may contribute to variability.

2.1 Simulator-to-itself variability (auto-inconsistency)

It may seem peculiar that the same LPC software applied on the same layout can give different results; random generators aside, software should be deterministic. But let us consider the situation in which the same physical design (such as a standard cell) is instantiated twice, once in the vertical position, and once in the lateral position, rotated by 90 degrees. Even assuming horizontal-vertical symmetry in the illumination, simulation results of these two instantiations may still be different if the lithography model exhibits asymmetries.

Careful model building may eliminate such model-induced fluctuation. Nevertheless, variability may remain if the image is computed using the so-called dense approach.[9] This method can be implemented by computing image intensities at a set of grid points. Intervening values are determined by interpolation from the intensities at these grid points. Imagine a scenario with a square image grid of spacing δ[§] and two instantiations of the same physical design having a relative translation of $(\Delta x, \Delta y)$. If Δx or Δy is not an integral multiple of δ, images of the two instantiations can be slightly different due to interpolation. Such discrepancy leads to variability in both LPC and OPC.

Measures for improving computation throughput may also contribute to variability. Consider the situation shown in Fig. 4(a), where a long wire terminates at a region densely populated with shapes, and our hotspot interest lies within this dense region. During hierarchical operation or parallel processing the long wire may be truncated to limit the area of computation. Suppose two instances of the layout are truncated at different locations indicated by the labels "case 1" and "case 2" in the figure. Let us further suppose the following rudimentary set of retargeting rules:

1. edges longer than $2 \times d_{\min}$ are segmented uniformly;
2. each segmented edge length lies between $d_{\min}$ and $d_{\max}$; and
3. the local bias each segmented edge receives is a function of spacing to its nearest neighbor.

The edges of the truncated long wire would then be segmented differently between "case 1" and "case 2," as shown in Fig. 4(b). Let's say the truncated original edge of "case 1" is segmented into five parts, as illustrated by the left edge of the long wire in Fig. 4(b); each segment is of length $d_{\min}$. At the same time, the shorter truncated original edge of "case 2" would have an altogether distinct segmentation of, say, three parts of $d_{\max}$ long each [depicted by the right edge of the long wire in Fig. 4(b)]. Such variance in segmentation, a result of minor differences in how a shape is truncated far away, directly translates into differing retargeted layouts, as shown in Fig. 4(c), which in turn causes variability in the lithography image, and hence OPC and LPC in the region of interest.

OPC iteration is another source of variability. Some OPC algorithms adjust edges sequentially such that designed masks exhibit a dependency on the order in which the edges are moved. It is possible to contrive a situation, as shown in Fig. 5, in which starting OPC iteration from opposite corners of a $40\,\mu\text{m} \times 40\,\mu\text{m}$ 65-nm layout would result in a simulated critical dimension discrepancy of more than 15% at the center.

[§]A square grid is chosen for exposition convenience here. In general the grid can be non-uniform and non-rectangular.

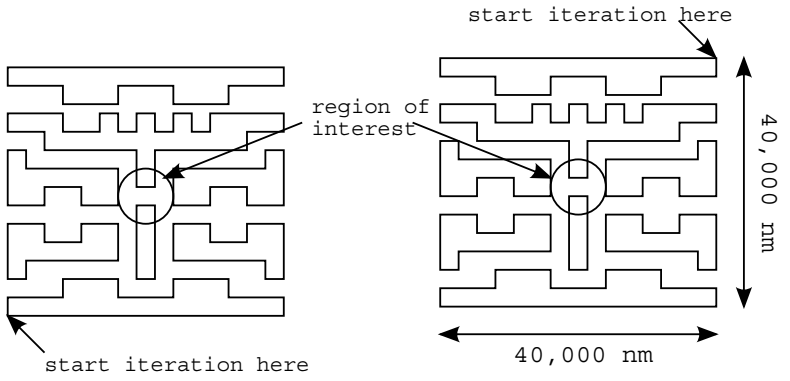


Figure 5. OPC result is dependent on the order in which edges are moved. Critical dimension discrepancy as large as 15% is possible.

2.2 Simulator-to-simulator variability (across-simulator mismatch)

During the hotspot analysis stage in physical DFM, it is sometimes impractical, for turn-around or for accessibility reasons, to apply the same algorithm as the one used for sign-off verification. The use of different algorithms leads to many more sources of inconsistency. Possible culprits include all aspects of hotspot computation, from model fitting to image calculation, from retargeting to OPC, from contouring to deriving process robustness. Even when two algorithms meet the same set of specifications, implementation differences may still result in variability.

As an example, consider image calculation given a set of coherently decomposed kernels.[10] Assuming the kernel set is discretized, i. e., each kernel is specified as a collection of values at discrete locations, any algorithm would need to interpolate these supplied values during image calculation.[¶] From accuracy considerations, such a feat is best performed using the sinc sampling function.[11] For throughput concerns, however, a relatively local scheme such as cubic spline may be expedient. Algorithms making different accuracy-throughput trade-offs would give discrepant images even from the same specified kernel set.

3. SIMULATOR MATCHING

3.1 Formulation

Existence of variability in computation lithography algorithms means that results from any application would not be perfectly accurate or absolutely precise. Consider an example hotspot analysis procedure that comprises the following three steps:[12]

1. identify candidate hotspots from layout;
2. compute critical dimension of each candidate; and
3. classify each candidate as hotspot or non-hotspot according to calculated critical dimension.

Applying this three-step procedure, the critical dimensions of all candidates of a layout would form a distribution. Let $\mathcal{P}_{\text{noiseless}}(x)$ denote the true distribution. It may resemble the solid curve in Fig. 6; but we have no way of knowing unless we have a noiseless algorithm at our disposal. What we observe instead is $\mathcal{P}_{\text{variability}}(x)$, a tainted version of $\mathcal{P}_{\text{noiseless}}(x)$ that may look like the dotted curve in Fig. 6. If we assume algorithm variability is random, and that such randomness can be represented by the noise distribution function $f_{\text{noise}}(x)$, we can then establish the following expression:

$$\mathcal{P}_{\text{variability}}(x) = \mathcal{P}_{\text{noiseless}}(x) \star f_{\text{noise}}(x), \tag{1}$$

where $\star$ denotes convolution.

[¶]This interpolation is distinct from the interpolation for dense image computation discussed earlier in §2.1.

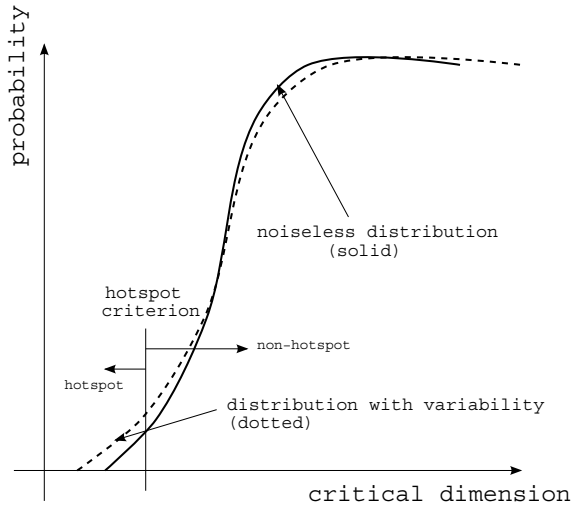


Figure 6. Example probability distribution of simulated critical dimension without noise (solid line) and with algorithm variability (dotted line).

Although Eq. (1) relates the noisy distribution to the true distribution, it does not enlighten us in practical situations. Of interest is the situation in which two analyses were preformed, resulting in two noisy distributions $\mathcal{P}_1(x)$ and $\mathcal{P}_2(x)$. These two distributions are related to $\mathcal{P}_{\text{noiseless}}(x)$, via an expression similar to Eq. (1) with noise functions $f_1(x)$ and $f_2(x)$ respectively. If the two analyses were performed using different algorithms, $f_1(x)$ and $f_2(x)$ are most likely different functions; this is the situation of across-simulator mismatch discussed in §2.2. One the other hand, if the two analyses were performed using the same algorithm (through, for example, differences in hierarchy management settings), $f_1(x) = f_2(x) = f_{\text{noise}}(x)$; this represents the situation of auto-inconsistency discussed in §2.1.

Regardless of auto-inconsistency or simulator mismatch, the two sets of hotspots from the two analyses are unlikely to be identical[‖]. For each candidate hotspot, taking into account the "truth"[**] as well as reported results from the two analyses, there are altogether eight possibilities, as shown in Table 3.1. They are labeled from (A) to (H) for ease of reference in subsequent discussions. For cases (B), (C), and (D), depending on whether the first or second algorithm is used, a true hotspot will go undetected and hence unremedied. On the other hand, extraneous fixing may occur for cases (F), (G), and (H). It is also of interest to note that, even if we perform LPC with two algorithms to safeguard against variability, in case (E), both analyses decide incorrectly that the same candidate is a hotspot. And we may be led to expending resource unnecessarily to rectify the extraneous hotspot. On the other hand, in case of (D), both analyses decide incorrectly that the same candidate is not a hotspot. We may be blissfully unaware until we receive our wafers!

Given a simulated critical dimension x, let us define our hotspot criterion as

$$x \quad \begin{cases} < T & \text{classified as hotspot,} \\ \geq T & \text{classified as non-hotspot,} \end{cases}$$

[‖]Note that $f_1(x) = f_2(x)$ implies $\mathcal{P}_1(x) = \mathcal{P}_2(x)$. But this does not mean that the set of hotspots (with critical dimensions less than the threshold T) in $\mathcal{P}_1(x)$ and $\mathcal{P}_2(x)$ are the same.

[**]We put "truth" in quotes here because "truth" is something of which we have no knowledge. But this does not prevent examination of correlation between the two noisy distributions as derived below.

Table 1. Eight cases of hotspot determination from two analyses with variability. A $\surd$ means a particular hotspot classification agrees with the truth; a $\times$ means the reverse.

Label	Truth	Simulator 1	Simulator 2
(A)	Hotspot	$\surd$	$\surd$
(B)	Hotspot	$\surd$	$\times$
(C)	Hotspot	$\times$	$\surd$
(D)	Hotspot	$\times$	$\times$
(E)	Non-hotspot	$\surd$	$\surd$
(F)	Non-hotspot	$\surd$	$\times$
(G)	Non-hotspot	$\times$	$\surd$
(H)	Non-hotspot	$\times$	$\times$

where T is the hotspot threshold. We can then derive the occurrence probability of the eight cases tabulated in Table 3.1 as follows:

$$
\begin{aligned}
(A): &\quad \mathcal{P}_A = \int_{-\infty}^{T} \mathcal{P}_{\text{noiseless}}(x) g_1(T-x) g_2(T-x)\,\mathrm{d}x,\\
(B): &\quad \mathcal{P}_B = \int_{-\infty}^{T} \mathcal{P}_{\text{noiseless}}(x) g_1(T-x)[1-g_2(T-x)]\,\mathrm{d}x,\\
(C): &\quad \mathcal{P}_C = \int_{-\infty}^{T} \mathcal{P}_{\text{noiseless}}(x)[1-g_1(T-x)] g_2(T-x)\,\mathrm{d}x,\\
(D): &\quad \mathcal{P}_D = \int_{-\infty}^{T} \mathcal{P}_{\text{noiseless}}(x)[1-g_1(T-x)][1-g_2(T-x)]\,\mathrm{d}x,\\
(E): &\quad \mathcal{P}_E = \int_{T}^{\infty} \mathcal{P}_{\text{noiseless}}(x)[1-g_1(T-x)][1-g_2(T-x)]\,\mathrm{d}x,\\
(F): &\quad \mathcal{P}_F = \int_{T}^{\infty} \mathcal{P}_{\text{noiseless}}(x)[1-g_1(T-x)] g_2(T-x)\,\mathrm{d}x,\\
(G): &\quad \mathcal{P}_G = \int_{T}^{\infty} \mathcal{P}_{\text{noiseless}}(x) g_1(T-x)[1-g_2(T-x)]\,\mathrm{d}x,\\
(H): &\quad \mathcal{P}_H = \int_{T}^{\infty} \mathcal{P}_{\text{noiseless}}(x) g_1(T-x) g_2(T-x)\,\mathrm{d}x,
\end{aligned}
$$

where

$$
g_{1,2}(x) = \int_{-\infty}^{x} f_{1,2}(x')\,\mathrm{d}x' \quad \text{is the cumulative noise distribution function, and}
$$

$$
\mathcal{P}_A + \mathcal{P}_B + \mathcal{P}_C + \mathcal{P}_D + \mathcal{P}_E + \mathcal{P}_F + \mathcal{P}_G + \mathcal{P}_H = 1.
$$

$(\mathcal{P}_A + \mathcal{P}_H)$ accounts for the fraction of hotspots common to both analyses; $(\mathcal{P}_B + \mathcal{P}_G)$ is proportional to the number of hotspots found in the first analysis but not in the second, and $(\mathcal{P}_C + \mathcal{P}_F)$ denotes the probability of a candidate determined to be a hotspot in analysis two but not in analysis one.

If we call the first analysis *reference* and the second *trial*, we can define the hotspot matching rate as

$$
\mathcal{P}_{\text{matching}} = \frac{\text{number of common hotspots}}{\text{number of hotspots in reference}} = \frac{\mathcal{P}_A + \mathcal{P}_H}{\mathcal{P}_A + \mathcal{P}_B + \mathcal{P}_G + \mathcal{P}_H},
$$

and the extra hotspot rate as

$$
\mathcal{P}_{\text{extra}} = \frac{\text{number of trial hotspots not in reference}}{\text{number of hotspots in reference}} = \frac{\mathcal{P}_C + \mathcal{P}_F}{\mathcal{P}_A + \mathcal{P}_B + \mathcal{P}_G + \mathcal{P}_H}.
$$

The missing hotspot probability is then

$$\mathcal{P}_{\text{missing}} = 1 - \mathcal{P}_{\text{matching}}.$$

3.2 Ramifications

Let us consider a few examples from the 65-nm technology that illustrate the consequence of algorithm variability in hotspot analysis. Assume a unit-less hotspot threshold $T = 100$, and that the hotspot distribution $\mathcal{P}_{\text{noiseless}}(x)$ around the threshold is a linearly increasing function of the critical dimension x, such as that shown in Fig. 6, with a slope of $1/7,200$ and an x-intercept of 80. Further suppose that variability is characterized by a Gaussian distribution, i. e., the noise functions $f_1(x)$ and $f_2(x)$ can respectively be described by the standard deviations σ_1 and σ_2, we evaluate the matching and extra rates of a few scenarios using numerical integration.

Case I: $\sigma_1 = \sigma_2 = 15$

A variation amounting to 15% of the hotspot threshold results in substantial mismatch between results from the two analyses. The matching rate is 43.9% while the extra rate is 56.1%. Note that

$$\mathcal{P}_{\text{missing}} = (1 - \mathcal{P}_{\text{matching}}) = \mathcal{P}_{\text{extra}}$$

in this case because of symmetry between the two analyses. The missing rate $(\mathcal{P}_B + \mathcal{P}_F)$ and the extra probability $(\mathcal{P}_C + \mathcal{P}_G)$ are equal because the two analyses have the same variability function.

Case II: $\sigma_1 = \sigma_2 = 5$

Compared with Case I, a smaller variability leads to better matching (73.4%) and less extra (26.6%). Nevertheless, at a variability level of 5% of the hotspot threshold, as many as one out of four hotspots are still mismatched. Note that we again have $\mathcal{P}_{\text{missing}} = \mathcal{P}_{\text{extra}}$ due to symmetry of the two analyses.

Case III: $\sigma_1 = \sigma_2 = 5$; increased trial threshold

If it is desirable to increase the matching rate given a fixed level of variability, we can increase the hotspot threshold of the trial analysis. Using the same variabilities as those in case II but increasing the hotspot threshold of the trial analysis to 105, i. e., $T_1 = 100$ and $T_2 = 105$, it is possible to increase the matching rate to close to 90% at a significantly higher extra percentage of over 60%.

Case IV: $\sigma_1 = 5; \sigma_2 = 15$

For the case in which variability of one analysis, say, the reference, is less than that of the second (the trial), the missing and extra rates are no longer the same. The results for $\sigma_1 = 5$ and $\sigma_2 = 15$ are tabulated in Table 3.2, along with the previous three cases for comparison. Note that the missing rate is less than the extra rate because the extra hotspots found by the first analysis $(\mathcal{P}_B + \mathcal{P}_G)$ are fewer than their counterparts in the second. This is a consequence of our assumption of an increasing distribution function $\mathcal{P}_{\text{noiseless}}(x)$ around the threshold.

In other words, under the assumption, which is often true in practice, of an increasing distribution function $\mathcal{P}_{\text{noiseless}}(x)$ around the hotspot threshold, the matching and extra rates can be used to determine the relative noise levels of algorithms. If $\mathcal{P}_{\text{missing}} < \mathcal{P}_{\text{extra}}$ we can infer that the reference is closer to the truth. The trial has less variability if the inequality is reversed.

4. IN FACE OF VARIABILITY

Variability should be addressed on both the algorithm and the application levels. For the former, we need to improve algorithmic precision via examination and control of the root causes of variability. This is essentially an effort to shape the noise function $f_{\text{noise}}(x)$ in §3.1 closer to a delta function:

$$f_{\text{noise}}(x) \longrightarrow \delta(x).$$

At the same time, for each application, we need to set specifications that are at once commensurate with the limitations of the underlying algorithms and the goals of the particular application. For example, in a physical DFM application in which the hotspot analysis algorithm is distinct from the sign-off verification engine, it is impractical to expect fixing of all hotspots, since, according to §3.2, unless we are willing to tolerate a large number of erroneous hotspots, 100% hotspot matching is highly unlikely.

Table 2. Numerical simulation of various cases of algorithm variability.

	Case I	Case II	Case III	Case IV
$\mathcal{P}_A$	0.0128	0.0201	0.0228	0.0160
$\mathcal{P}_B$	0.0062	0.0035	0.0008	0.0076
$\mathcal{P}_C$	0.0062	0.0035	0.0049	0.0030
$\mathcal{P}_D$	0.0038	0.0018	0.0005	0.0023
$\mathcal{P}_E$	0.9299	0.9612	0.9517	0.9441
$\mathcal{P}_F$	0.0176	0.0042	0.0137	0.0213
$\mathcal{P}_G$	0.0176	0.0042	0.0023	0.0034
$\mathcal{P}_H$	0.0059	0.0014	0.0034	0.0023
matching rate	43.9%	73.4%	89.4%	62.3%
extra rate	56.1%	26.6%	63.5%	82.9%

5. REMARKS

Computation lithography owes its current prevalence to algorithm innovations[13] and improvement in CPU performance over the last few decades. In this virtual world, if we define reality as applications characterized by

1. specifications that make engineering sense, and
2. performance that meets the specification targets,

i. e., tools that perform useful service for IC development, and describe virtuality as applications that fall short of one or both of the above two criteria, there is only a fine line between virtual reality and virtual virtuality.[14] An understanding of algorithm variability would help us delineate this boundary.

REFERENCES

[1] Shang, S. D., Granik, Y., Cobb, N. B., Maurer, W., Cui, Y., Liebmann, L. W., Oberschmidt, J. M., Singh, R. N., and Vampatella, B. R., "Failure prediction across process window for robust OPC," in [*Proc. SPIE*], Yen, A., ed., **5040**, 431–440 (June 2003).

[2] Kim, J. and Fan, M., "Hotspot detection on post-OPC layout using full-chip simulation-based verification tool: a case study with aerial image simulation," in [*Proc. SPIE*], Kimmel, K. R. and Staud, W., eds., **5256**, 919–925 (Dec. 2003).

[3] Wikipedia, "Design rule checking," (Oct. 2008). http://en.wikipedia.org/wiki/Design_rule_checking.

[4] Taravade, K. N., Croffie, E. H., and Jost, A., "Two-dimensional image-based model calibration for OPC applications," in [*Proc. SPIE*], Smith, B., ed., **5377**, 1522–1527 (2004).

[5] Mansfield, S., Han, G., and Liebmann, L., "Through-process modeling for design-for-manufacturability applications," *Journal of Microlithography, Microfabrication, and Microsystems* **6**, 031007 (Sept. 2007).

[6] Zhang, Y., Feng, M., and Liu, H.-Y., "A focus exposure matrix model for full chip lithography manufacturability check and optical proximity correction," in [*Proc. SPIE*], Hoga, M., ed., **6283**, 62830W (Apr. 2006).

[7] Wong, A. K., Molless, A. F., Brunner, T. A., Coker, E., Fair, R. H., Mack, G. L., and Mansfield, S. M., "Linewidth variation characterization by spatial decomposition," *Journal of Microlithography, Microfabrication, and Microsystems* **1**, 106–116 (July 2002).

[8] Wong, A. K., [*Resolution Enhancement Techniques in Optical Lithography*], 55–58, SPIE Press (2001).

[9] Cobb, N. B. and Granik, Y., "Dense OPC for 65nm and below," in [*Proc. SPIE*], Weed, J. T. and Martin, P. M., eds., **5992**, 599259 (2005).

[10] Pati, Y. C., Ghazanfarian, A. A., and Pease, R. F., "Exploiting structure in fast aerial image computation for integrated circuit patterns," *IEEE Transactions on Semiconductor Manufacturing* **10**, 62–74 (Feb. 1997).

[11] Gamo, H., "Matrix treatment of partial coherence," in [*Progress in Optics*], Wolf, E., ed., **3**, 187–332, North-Holland (1964).

[12] Kong, T., Leung, H., Raghavan, V., Wong, A., and Xu, S., "Model-assisted routing for improved lithography robustness," in [*Proc. SPIE*], Wong, A. K. K. and Singh, V. K., eds., **6521**, 65210D (Mar. 2007).

[13] Neureuther, A., "If it moves, simulate it!," in [*Proc. SPIE*], Levinson, H. J. and Dusa, M. V., eds., **6924**, 692402 (2008).

[14] Wong, A. K. and Lam, E. Y., "Computation lithography: virtual reality and virtual virtuality," in [*ECS Transactions—7th ISTC/CISC*], **12** (Mar. 2009).