Microelectronic Engineering 168 (2016) 15–26

Contents lists available at ScienceDirect

Microelectronic Engineering

journal homepage: www.elsevier.com/locate/mee

Research paper

A fast and manufacture-friendly optical proximity correction based on machine learning

Xu Ma[a], Shangliang Jiang[b], Jie Wang[a], Bingliang Wu[a], Zhiyang Song[a], Yanqiu Li[a,*]

[a] Key Laboratory of Photoelectronic Imaging Technology and System of Ministry of Education of China, School of Optoelectronics, Beijing Institute of Technology, Beijing 100081, PR China
[b] Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, CA 94706, U.S.A

ARTICLE INFO

Article history:
Received 11 April 2016
Accepted 10 October 2016
Available online 18 October 2016

Keywords:
Optical lithography
Optical proximity correction
Machine learning
Nonparametric kernel regression
Manufacturability

ABSTRACT

Pixel-based optical proximity correction (PBOPC) is currently a key resolution enhancement technique to push the resolution limit of optical lithography. However, the increasing scale, density and complexity of modern integrated circuits pose new challenges to both of the OPC computational intensity and mask manufacturability. This paper aims at developing a practical OPC algorithm based on a machine learning technique to effectively reduce the PBOPC runtime and mask complexity. We first divide the target layout into small regions around corners and edge fragments. Using a nonparametric kernel regression technique, these small regions are then filled in by the weighted linear combination of a subset of training OPC examples selected from the pre-calculated libraries. To keep balance between the image fidelity and mask complexity, we use an edge-based OPC (EBOPC) library to synthesize the OPC patterns in non-critical areas, while use another PBOPC library for hotspots. In addition, a post-processing method is developed to refine the regressed OPC pattern so as to guarantee the final image fidelity and mask manufacturability. Experimental results show that, compared to a currently professional PBOPC software, the proposed algorithm can achieve approximately two-fold speedup and more manufacture-friendly OPC patterns.

1. Introduction

The electronics industry has relied on resolution enhancement techniques (RET) to enhance the imaging performance of optical lithography systems [1–4]. As one of key RETs, optical proximity correction (OPC) method pre-warps the masks to compensate for imaging distortions as the target patterns are replicated onto semiconductor wafers. In general, OPC approaches can be classified into rule-based OPC and model-based OPC [2]. Rule-based OPC is mostly heuristic and simple to implement, but not competent for the technology nodes beyond 90 nm. In contrast, model-based OPC uses physical or mathematical models to formulate the OPC framework and seeks the global optimal solution, which may further push the lithographic resolution limit. Model-based OPC approaches include edge-based OPC (EBOPC) and pixel-based OPC (PBOPC). EBOPC decomposes the edges of mask into segments and gradually nudges them to find the optimal locations, whereas PBOPC grids the mask into small pixels and optimizes their transmission coefficients [2]. Although EBOPC usually obtains much simpler mask patterns than PBOPC, it may not be very suitable for 45 nm and smaller nodes due to the lack of degrees of optimization freedom [5,6]. On the other hand, PBOPC overcomes this drawback, and may result in higher image fidelity on the wafer. Consequently, a set of PBOPC approaches have been proposed for advanced optical lithography to enhance the imaging performance at nominal settings [7–15] or over a range of process variations [16–22].

In general, model-based OPC needs to process a mass of data and carry out time-consuming calculations, especially for the current sophisticated large scale layout. To alleviate this problem, different machine learning techniques were applied to effectively accelerate the OPC design process [23–25]. In the past, Gu, Zakhor and Gao used linear regression and principle component regression methods to estimate the movement directions of edge and corner fragments, and effectively reduced the iterations required for the EBOPC algorithms [26,27]. By treating all mask pixels as optimization variables, PBOPC approaches introduce new challenges to both computational efficiency and mask manufacturability. Consequently, Luo proposed a fast PBOPC method using the multilayer perceptron neural network [28]. Luo and Shi, et. al. developed a support vector machine (SVM) based layout retargeting method to promote the convergence speed of PBOPC optimization process [29]. Recently, Ma and Li, et al. proposed a fast PBOPC method based on nonparametric kernel

* Corresponding author.
E-mail address: liyanqiu@bit.edu.cn (Y. Li).

http://dx.doi.org/10.1016/j.mee.2016.10.006
0167-9317/

regression to effectively speedup the iterative PBOPC flow on 90 nm and 45 nm Metal layers [30]. However, how to further improve the mask manufacturability in this method still needs to be investigated, which is a key point to be concerned in advanced lithography technology nodes. In addition, the impact of parameter selection on the algorithm performance needs further study, and this method is yet to be proven by more dense layout, such as poly layer mask pattern.

In order to overcome the limitations of the prior work in [30], this paper focuses on developing a fast and manufacture-friendly OPC approach based on nonparametric kernel regression to effectively speedup the currently professional PBOPC software and ameliorate the mask manufacturability. The professional software used in this paper is Calibre [31]. The Calibre nmOPC package and Calibre pxOPC package provide the functions to calculate EBOPC and PBOPC. Hereafter, the Calibre nmOPC and Calibre pxOPC are referred to as the "EBOPC software" and "PBOPC software", respectively. In practice, any other OPC approaches or software could be applied. The basic idea of this paper is to rapidly synthesize a raw OPC pattern based on the pre-calculated EBOPC and PBOPC libraries to effectively accelerate the OPC optimization process. In addition, the proposed method takes the advantages of both EBOPC and PBOPC methods to keep balance between the imaging performance and mask complexity. The flowchart of the proposed method is shown in Fig. 1. We first use the professional software to calculate the EBOPC and PBOPC results of a training layout. Based on these results, the EBOPC and PBOPC libraries are built up to provide apriori-knowledge for the following regression process. Subsequently, the test layout to be optimized is divided into three kinds of regions: convex corners, concave corners and edge fragments. These regions are then filled in by the pre-calculated OPC pieces selected from the two libraries via using the nonparametric kernel regression technique. Particularly, we use the EBOPC library for the non-critical areas that only need simple mask corrections. On the other hand, we use the PBOPC library for the hotspots, where careful mask corrections and additional sub-resolution assist features (SRAF) are required. By properly applying different degrees of mask corrections according to the local characteristics of mask features, we are able to get preferable mask manufacturability than merely using PBOPC method. Finally, a post-processing method is proposed to refine the regressed OPC pattern so as to guarantee the image fidelity on wafer and mask manufacturability. Following sections will describe each step of the flowchart in detail. The proposed algorithm is tested using a Metal layer and another Poly layer at 45 nm technology node. Simulations show that, compared to the PBOPC software, the proposed method can achieve approximately two-fold speedup and more manufacture-friendly OPC pattern. At the end of this paper, we investigate and analyze the influence of two key parameters on the algorithm performance. One parameter is the number of candidates chosen from libraries to synthesize each OPC piece, and the other is the threshold value used to switch between EBOPC and PBOPC libraries. Different from this paper, our prior work in [30] was based on PBOPC libraries only, the impact of key parameter values has not been studied, and the algorithm has not been tested on the Poly layers yet.

The remainder of this paper is organized as follows: The nonparametric kernel regression technique is briefly discussed in Section 2. The method to construct the OPC libraries is described in Section 3. The fast and manufacture-friendly OPC algorithm based on nonparametric kernel regression is developed in Section 4. Simulations and discussions are presented in Section 5. Conclusions are provided in Section 6.

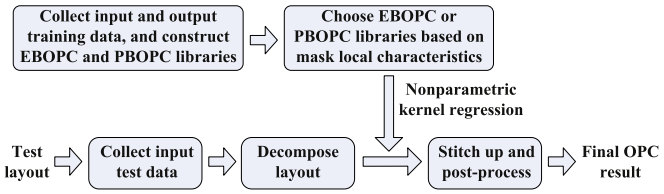

Fig. 1. Flowchart of the proposed OPC approach.

2. Nonparametric kernel regression

Suppose there are twosets of data $\vec{x}$ and $\vec{y}$ having the following relationship $\vec{y} = f(\vec{x}) + \vec{\epsilon}$, where $f(\cdot)$ is an arbitrary function, and $\vec{\epsilon}$ is an error vector. Nonparametric regression is a statistical technique, which models the dependence of the output $\vec{y}$ on the input features $\vec{x}$. Nonparametric regression methods are suitable to model general nonlinear patterns hidden in a high-dimensional data space, where $f(\cdot)$ is assumed to be a continuous function with unknown form [32,33]. On the other hand, OPC synthesis is an ill-posed nonlinear problem in high-dimensional data space, where all mask pixels or edge segment locations can be treated as optimization variables. Thus, the nonparametric regression technique is suitable to solve for the OPC problem.

A variety of nonparametric regression approaches have been investigated in literature [34,35]. In this paper, we adopt the kernel-based nonparametric regression method in our algorithm. The Nadaraya-Watson kernel regression method was independently developed by Nadaraya and Watson, which takes the general form [36,37]

$$\hat{\vec{y}}_t = \hat{f}(\vec{x}_t) = \frac{\sum_{i=1}^{N} \vec{y}_i K(\vec{x}_t, \vec{x}_i)}{\sum_{i=1}^{N} K(\vec{x}_t, \vec{x}_i)}, \tag{1}$$

where $\vec{x}_i$ and $\vec{y}_i$ are the input and output training data at hand, $\vec{x}_t$ and $\vec{y}_t$ are the input and output test data, $\hat{\vec{y}}_t$ is the estimate of $\vec{y}_t$, N is a tunable parameter representing the candidate number selected from the training data set, and $K(\vec{x}_t, \vec{x}_i)$ is the kernel function. The kernel function $K(\vec{x}_t, \vec{x}_i)$ measures the similarity between the observations at $\vec{x}_i$ and a given location $\vec{x}_t$, and weights each $\vec{y}_i$ to predict $\vec{y}_t$ [34]. In this paper, we choose the Gaussian kernel

$$K(\vec{x}_t, \vec{x}_i) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left\|\frac{\vec{x}_t - \vec{x}_i}{h}\right\|_2^2\right), \tag{2}$$

where h is the bandwidth to control the smoothing range. The underlying rationale of applying nonparametric kernel regression to OPC problem is that the similar local features on target pattern likely correspond to similar OPC solutions. Thus, we can approximate the OPC solution of test layout by weighted averaging several training OPC examples corresponding to the target patterns that resemble the test layout. Particularly, in the OPC problem $\vec{x}_i$ represents the vector of the sampling points around a certain mask feature on the training layout, $\vec{y}_i$ is the optimized OPC pattern corresponding to $\vec{x}_i$. A mass of $(\vec{x}_i, \vec{y}_i)$ combinations are precalculated and saved in the OPC libraries. Given a test layout to be optimized, $\vec{x}_t$ represents the vector of sampling points around a mask feature on the test layout, and $\hat{\vec{y}}_t$ is the regressed OPC pattern corresponding to $\vec{x}_t$. According to Eq. (1), N candidates are selected from the libraries that have the smallest distance of $\|\vec{x}_t - \vec{x}_i\|_2^2$ among all training data. The construction method of the OPC libraries is described in the next section.

3. Construction of OPC libraries

3.1. Collection of input training data

The input training data $\vec{x}_i$ in Eq. (1) represents the local geometry characteristic of the underlying mask pattern. In this paper, we use a Metal layer and another Poly layer at 45 nm technology node to