J SYST SCI SYST ENG
Vol. 32, No. 3, June 2023, pp. 320–351
DOI: https://doi.org/10.1007/s11518-023-5564-x
ISSN: 1004-3756 (paper), 1861-9576 (online)
CN 11-2983/N

DTFab: A Digital Twin based Approach for Optimal Reticle Management in Semiconductor Photolithography

Chandrasekhar Komaralingam Sivasubramanian,[a] Robert Dodge,[a] Aditya Ramani,[a] David Bayba,[b] Mani Janakiram,[b] Eric Butcher,[b] Joseph Gonzales,[b] Giulia Pedrielli[a]

[a]School of Computing and Augmented Intelligence, Arizona State University, Tempe, AZ, USA
ckomaral@asu.edu, rwdodge@asu.edu, aramani3@asu.edu, gpedriel@asu.edu (✉)
[b]Intel Corporation, Chandler, AZ, USA
david.m.bayba@intel.com, mani.janakiram@intel.com, eric.j.butcher@intel.com, joseph.b.gonzales@intel.com

Abstract. Photolithography is among the key phases in chip manufacturing. It is also among the most expensive with manufacturing equipment valued at the hundreds of millions of dollars. It is paramount that the process is ran efficiently, guaranteeing high resource utilization and low product cycle times. A key element in the operation of a photolithography system is the effective management of the reticles that are responsible for the imprinting of the circuit path on the wafers. Managing reticles means determining which are appropriate to mount on the very expensive scanners as a function of the product types being released to the system. Given the importance of the problem, several heuristic policies have been developed in the industry practice in an attempt to guarantee that the expensive tools are never idle. However, such policies have difficulties reacting to unforeseen events (e.g., unplanned failures, unavailability of reticles). On the other hand, the technological advance of the semiconductor industry in sensing at system and process level should be harnessed to improve on these "expert policies". In this manuscript, we develop a system for the real time reticle management that not only is able to retrieve information from the real system, but also is able to embed commonly used policies to improve upon them. We develop a new digital twin for the photolithography process that efficiently and accurately predicts the system performance, thus allowing our system to make predictions for future behaviors as a function of possible decisions. Our results demonstrate the validity of the developed model, and the feasibility of the overall approach demonstrating a statistically significant improvement of performance as compared to the current policy.

Keywords: Semiconductor manufacturing, reinforcement learning, reticle management, digital twin

1. Introduction

Within silicon-based semiconductor manufacturing, the equipment required for the photolithography process is by far the most expensive single piece of machinery in the process, with typical machines costing in the tens to hundreds of millions of dollars (Byrne 2007). These machines utilize opaque reticles (also referred to as masks) to imprint the different layers of integrated circuits that form a microchip onto silicon wafers. These layers form the core of a microchip, thus making the photolithography process among the most critical phases for the successful manufacturing of the end product. In fact, the tool responsible for the photolithography is among the most expensive in a semiconductor factory (Sterling 2022, Schoolov 2022). ASML Holding, one of the major producers of photolithography

equipment, as of January 2022, was selling its most advanced machines in current commercial production, known as EUV lithography systems because of the "Extreme Ultraviolet" light waves they use to map out the circuitry of computer chips, for around $150 million each (Sterling 2022). Among the customers are the largest semiconductor companies, whose mission is to optimize the productivity of such expensive resources. In particular, a successful management of the photolithography phase maximizes the utilization of the photolithography machines, while guaranteeing low product cycle times and low defects. While the last aspect is mainly managed through maintenance policy, the first two are highly impacted by the operational strategies applied on the shop floor.

As previously mentioned, in the photolithography process, wafers undergo a sequence of operations responsible for imprinting the pattern of a circuit layer. To achieve the correct print, reticles are used by manufacturing machines, scanners, as step-and-repeat aligners to transfer a pattern image of the integrated circuit onto the wafer (EESEMI 2005, Byrne 2007). Figure 1 shows the principle underlying the wafer printing through a reticle. As it can be observed, the process works in principle like a photo-negative, where a light is shone through a mask containing the circuit pattern, after which it goes through a series of projection lenses for de-magnification before reaching the wafer. This pattern transfer usually covers a small portion of the wafer. As a result, the process will need to be repeated multiple times until the entire wafer is covered before moving onto the next wafer (Wikichip 2023).

Given the cost of the scanner, a major operational imperative is to keep it utilized as much as possible. This can be achieved with a large number of reticles available for the tool so that it can process continuously over several patterns. While this in principle can address utilization, it may hurt cycle times if the reticles are not properly assigned to the tools. As suggested in Park et al. (1999), on average, a photolithography process will have hundreds different product types, each requiring tens of unique reticles due to the difference in circuit designs. As a result, a realistic system may have thousands of reticles to be managed at each point in time. Hence, optimizing the allocation of reticles is all but a simple problem that, if handled improperly, creates an expensive bottleneck for the entirety of the manufacturing process (Benzoni et al. 2020, Byrne 2007, Peters and Puharic 2003, Vitelli 2021). As a result, the *reticle management policy* is a very important component of the photolithography management that determines how reticles are allocated to the tools during production. Compounding the issue of high dimensionality is the need to setup reticles, i.e., inspect the quality of the designs when they start to be adopted in line, and the uncertainty due to unexpected failures of the resources (Carranza 1986, Allgeier et al. 2020).

Background and related approaches. While the reticle management problem is unique to semiconductors, it can be interpreted as a tool management problem in traditional manufacturing domains. In this sense, approaches to solve this class of problems have traditionally fallen into three categories: (i) formulating

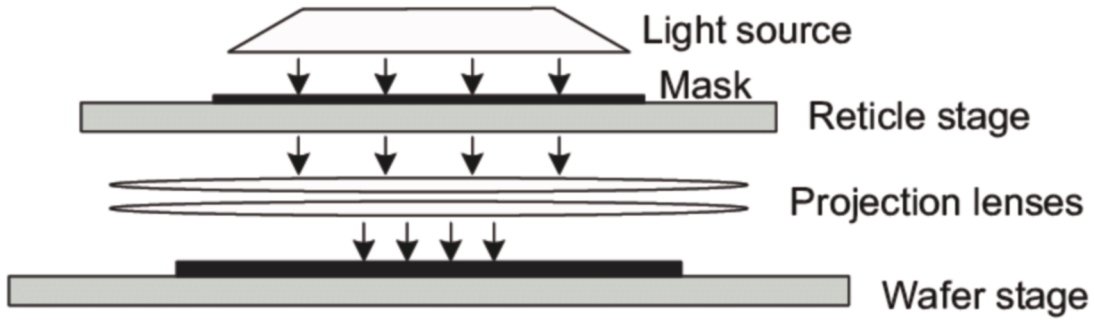


Figure 1 Schematic of the photolithography process[1]

[1] Source: Sun L, Chen X, Tomizuka M(2014). Selective iterative learning control to deal with iteration-dependent disturbance. In *Proceedings of ISCIE/ASME International Symposium on Flexible Automation* (pp. 1-8) (Sun et al. 2014).

the problem as an optimal schedule, mathematical programming is used to formulate and solve the problem; (ii) heuristic policies, very common in industry practice, do not typically require expensive optimization tools, but tend to be sub-optimal; (iii) the problem has been formulated as optimal control, attempting to find the state dependent policy. In this area, reinforcement learning has gained increasing attention. *Mathematical programming* has been used extensively to solve tool allocation problems in general flexible Manufacturing Systems (FMS), and similar approaches can be found in photolithography operations scheduling (Turkcan et al. 2003, Turkcan 2007). Therein, a deterministic mathematical model is solved for an optimal production schedule constrained by tool and equipment availability (De Diaz 2005). Approaches that rely on *heuristics rules* for the allocation of reticles are very common in the industry practice and they have been extensively developed in the tool literature in manufacturing (Hung and Chen 1998, Li et al. 2014, Fathi and Barnette 2002). These rules typically aim to create a near optimal product dispatching policy that maximizes the throughput of the system (Hung and Chen 1998, Li et al. 2014). While, in principle, a plethora of such rules can be developed, their performance is highly sensitive to process changes (Randhawa and Kuo 1997, Randhawa and Zeng 1996). Finally, tool management has been modeled as an *optimal control problem*. In particular, reinforcement learning has been recently adopted and analyzed as a solution approach. Specifically, approximately optimal policies are designed where dispatching decisions are made sequentially using information gained from the previous dispatch. To do so, trained reinforcement learning decision agents (typically a neutral network or Deep-Q-Learning agent) estimate the reward landscape at each step and take the action that is predicted to maximize the model's throughput. Examples of this approach are Waschneck et al. (2018), Park et al. (2020).

Contribution. In light of these challenges and the high value of the tools, it comes with no surprise that the semiconductor industry has led the development of Industry 4.0 technologies, enhancing sensing, and increasingly relying on ever improving digital copies of the highly

complex systems and processes happening in the Fab, with key players such as Siemens and Dassault Systèmes making the development of digital twin a key asset (Simens 2022, Systèmes 2021). This calls for a new way to approach challenges in the optimization of Fab operations with methods that can leverage novel technologies. However, most of the approaches do not leverage detailed simulations of the system, but, rather, focus on building approximations of the reward of interest. This is not aligned with the increasing availability of digital copies of complex systems and processes, as well as the plethora of information that can be gathered from resources in real time. On the other hand, heuristic policies developed by industry experts should be accounted for when generating potentially new approaches. However, the approaches in the literature do not have a way to embed policies to guide the search.

Our collaboration with Intel Corporation resulted in a novel approach to real time reticle management, which we propose in this work. Specifically, we propose a digital twin in the loop method for reticle management that uses a rollout algorithm to sequentially improve on a base policy for the allocation of reticles to the tools in the system. The method is able to take a decision that leads to the reduction of the product lead time each time a reticle is potentially up for change. We not only use the information on the state of the system that is available in real time, but develop a digital twin that efficiently and effectively evaluates the effect of the reticle allocation choice. Our results show the accuracy of the digital twin, and demonstrate that the new approach achieves a statistically improved performance over the base heuristic policy for reticle change.

Organization of the paper. The paper is organized as follows: Section 2 presents the relevant literature for the optimal reticle management problem; Section 3 describes the reticle management problem; Section 4 presents our approach, and Section 5 shows the main empirical results obtained. Finally, concluding remarks,and potential avenues for further work are discussed in Section 6.

2. Literature Review

Optimal reticle management can be seen within the broader class of policies for tool management in the semiconductor industry, and, more in general, for Flexible Manufacturing Systems (FMSs). In fact, several elements of the reticle management problem can be identified for many FMSs. Examples are the central tool storage, identical machines with tool magazines, and incoming products requiring a series of predetermined operations. As a result, in this section we review methods developed in the context of tool management in FMSs. We then highlight shortcomings with respect to the application of state-of-the-art methods to our problem. We categorize the proposed approaches into two main methodological classes: 1) Static optimization formulations that rely on exact mathematical programming formulations to produce state-independent solutions. For these approaches, the problem of tool management is solved through the use of deterministic mathematical modeling. 2) Simulation driven state-based approaches. Within this class, simulation models are used to model system behav-

iors and uncertainty through sequential state transitions. These simulation models are typically partnered with a machine learning algorithm or a decision heuristic that creates a decision policy using information from the current state.

Digital Twins vs Simulation. An important observation to make is the distinction between simulation and digital twin. Although both involve the implementation and simulation of what is essentially a virtual version of the given process, what separates the two is the ability of a digital twin to collect current state information of the physical system through sensors and perform optimization to the physical system in real-time (TWI Global 2022, Raghunathan 2019). Simulation models are limited due to the possible manual data collection and difficulty in data interpretation which would affect the timeliness of any necessary updates needed to be made to the physical system (TWI Global 2022, Raghunathan 2019). This therefore limits the scope where simulation models could be used to solve problems (TWI Global 2022). This is where Digital Twins could help fill the gap, where the real-time data is directly inputted into a virtual model which enables users to test different decision scenarios with the current state of the system and ensure that the best possible decision is selected at the appropriate time (Raghunathan 2019).

2.1 Static Optimization Approaches for Tool Management

In the context of complex manufacturing systems, the optimal tool management problem is identical to the job assignment problem (Fathi and Barnette 2002). As such, many static optimization problems have been formulated in the context of optimal tool management considering (Tang and Denardo 1998, Zeballos 2010, Turkcan 2007, Klemmt et al. 2010, Ham and Cho 2015, Bixby et al. 2006, Turkcan et al. 2003). In these implementations, optimal is usually defined as a minimization of cost or a maximisation of throughput. The constraints are typically each part requiring a particular tool, each machine only processing one part at a time, each station having a limited queue capacity, each station having a limited tool capacity, and each part requiring completion by specific time (Tang and Denardo 1998, Zeballos 2010, Turkcan et al. 2003, Turkcan 2007). Mathematical programming is also adopted in the industry practice, where work plans are prepared daily according to the modelled optimal plan. The plan is then changed by the line operators as a result of changes in the system (e.g., machine failure, operator unavailability) (Bradley et al. 1977). Methods differ on the considered constraints, as well as on the performance to be optimized. As an example, the model implemented in Tang and Denardo (1998) utilizes a deterministic model to minimize tool switches in a flexible manufacturing system by returning to storage the tool with the lowest demand among known future products. For this problem, they use the sequence in which jobs are processed and the set of tools placed on each machine prior to each job as their decision variables. These decisions are constrained by the capacity of each machine and the quantity of each tool. Similarly, Turkcan et al. (2003), Turkcan (2007) utilize a deterministic model to optimize a composite objective function of cost and tardiness. In this

problem, the decision variables are the operation parameters for each job, job assignments, and tool assignments. Constraining these decisions is the capacity of each station's tool magazine and queue and the life of each tool. They, along with Tang and Denardo (1998), propose a tool switching rule to aid in the minimization of switches in their systems. Zeballos (2010) solves a tool management problem in a deterministic FMS system in the context of four objective functions, minimization of make-span, minimization of tardiness, minimization of cost, and a multi-objective function combining all three. To optimize this function, this model decides an optimal tool and part allocation schedule. This schedule is constrained by machine capacity and tool quantity. Similar to Zeballos (2010), Turkcan et al. (2003), Turkcan (2007), Bixby et al. (2006) solves a deterministic tool management problem through mixed integer programming. In this model, the throughput of the model is minimized through the scheduling of product. This is constrained by the capacity of each station's tool magazine and queue. Klemmt et al. (2010) uses this optimal reticle management problem as part of a multistage problem, where tool and resist quantities are solved in stage one and used as the constraints of the operational tool management problem in stage two. In this model, load balancing and throughput are both used as the overall objective. For the tool management optimization, the problem is constrained by the tool composition purchased in stage one and the capacity of machines and the decision variable is the production schedule. A similar approach is demonstrated in Ham and Cho (2015), where a deterministic mathematical model is combined with an automated lot dispatching policy to optimize a composite objective function. In this system, the mathematical model solves for an optimal distribution of lots to be assigned to each machine. This is done through minimizing a composite objective of load balancing and cycle time. After the model is solved, the automated dispatch policy determines the order in which lots are sent to their assigned machines. In an attempt to account for the difficulty of capturing uncertainty using this type of model (Mishra et al. 2006), Rai et al. (2002) utilize a fuzzy goal approach to solve the deterministic mathematical model. In these approaches, they optimize the tool and part allocation problem to minimize the cost of the policy in an FMS system using fuzzy goal optimization. This optimization is constrained by tool allocations and machine capacity.

A major challenge of the presented approaches resides in the fact that the mathematical models used are closed-form in nature and computationally expensive. Fox (1983) presents the job shop scheduling programming model to be an NP-hard problem. Attempting to adapt a full manufacturing process to fit within the bounds of a deterministic mathematical model is difficult to do without the model becoming exceedingly large. Tang and Denardo (1998) summarizes the issue best, with these models, it may become necessary to make important assumptions regarding certain model parameters or behaviors, which in turn may reduce the applicability of the solutions to real-world systems. For these reasons, this type of modeling is infeasible to meet the requirements of clients where solutions may

be required quickly. Therefore, an alternate method is required to perform this task.

2.2 Simulation Driven State-based Approaches for Tool Management

A different branch of the literature investigates the performance of different policies, rather than designing them. To do this, simulation models are designed to mimic realistic conditions that are then used to test the performance of varying policies. Several examples of this in the field of semiconductor manufacturing, are De Diaz (2005), Hickie (1999), Park et al. (1999). In these papers, the photolithography phase of a semiconductor manufacturing process is simulated in a stochastic model to estimate the cycle time and load balancing of the system in different scenarios where factors such as product homogeneity, dispatching policy, forecasting, and station tool capacity are varied. Notably, De Diaz (2005) tests the impact of a reticle placement optimization subproblem on the performance of the simulation model. In this experiment, a programming model is used to solve for the optimal reticle placements on equipment in six-hour intervals. While De Diaz (2005) holds the most relevance to the optimal reticle management problem, the proposed approach is impractical due to its computational cost and long period of time required between decisions. The work in Hickie (1999), Park et al. (1999), on the other hand, deals with the model's cycle time and load balancing under varying dispatching policies similar to Randhawa and Zeng (1996), Randhawa and Kuo (1997). These approaches evaluate the throughput, cumulative tardiness, and number of tardy lots under the lot dispatching policies earliest due-date, lowest slack time, and shortest processing time. Several other basic dispatching rules are also examined by Hung and Chen (1998). They examine the effects of first-in-first-out, random, shortest remaining time, shortest queue for next operation, and shortest processing time policies have on a stochastic wafer manufacturing simulation's cycle time. Hung and Chen (1998), Leachman et al. (1988) propose and utilize a dispatching method on a stochastic simulation called the Queue Management policy. This policy uses information regarding the lots in queue at a station to decide the optimal time to release the next lot to that station. Li et al. (2014) proposes a newly designed lot dispatching rule to improve a stochastic semiconductor simulation model in three areas, cycle time, tardiness, and WIP. Unfortunately, these policies are static. Should the properties of a system change, a given rule-based policy may increase or decrease in effectiveness (Randhawa and Kuo 1997, Randhawa and Zeng 1996). We seek a policy that will be robust to future changes that may be made to the system, so this property is not ideal. The simulation executed in Byrne (2007), on the other hand, focuses on simulating two different photolithography stepper types, each using different reticle sets and running in a low volume, high product mix fab. The goal of this simulation is the optimization of costs via the determination of the amount of reticle coverage required in order to effectively balance utilisation and average lot wait times, and the results of which are used to set the appropriate reticle purchase plans. Another approach within this category is reinforcement learning.

Reinforcement learning aims to maximize the performance of a model with respect to a cumulative reward function. To do this, a decision agent sequentially chooses actions from an action space derived from the current state of the system. These approaches are almost always used with stochastic models. The contributions (Waschneck et al. 2018, Park et al. 2020, Kim et al. 2021) provide examples of this approach. In Waschneck et al. (2018), Park et al. (2020), the authors implement neural network decision agents, while Kim et al. (2021) implements Deep Q-Learning to make dispatching decisions in a semiconductor manufacturing process. In Waschneck et al. (2018), the authors utilize three neural network decision agents to make product dispatching decisions for three sequential processes based on the reward function of the cycle time. The work in Park et al. (2020) on the other hand, uses a single neural network dispatching agent whose goal is to minimize the makespan of the system. The work in Zhang and Dietterich (1995) is the earliest implementation of reinforcement learning to this type of problem. They utilize this technique to optimize dispatching a static job shop system in regards to a special reward function called the *resource dilation factor*. The contributions (Zhang et al. 2007 2012) address photolithography job scheduling with minimal tardiness solving a reinforcement learning formulation of the problem through Deep Q-Learning.

3. The Reticle Management Problem

Before defining the reticle management problem in mathematical terms, it is important to first understand the problem in an intuitive sense. As discussed in Section 1, each operation within the photolithography process requires a specific reticle to be used. This presents an interesting problem since a given fab may have hundreds to thousands of different reticles, while only being able to mount a select few for use at a given time. Because of this imbalance of reticles to tool space, some kind of decision guide must be implemented to manage which reticles are removed from tools and which reticles are brought out of storage between operations. To do this, we formulate this problem as a stochastic control problem leveraging the power of digital twin technology to evaluate potential reticle swaps. To begin, we present the key notations and definitions. Later in the paper, Section 3.2 presents the formal reticle management problem formulated as a stochastic optimal control problem, the computational challenges are highlighted and the simulation based rollout algorithm is justified in this context.

3.1 Notations and Definitions

The state of the system is $x_k = \{R_k, L_k, W_k\}$, referring to the reticles, lots, and wafers states. In particular:

$$R_k = \{r_{i,k}\}, r_{i,k} = \begin{cases} 0, & \text{if reticle in storage} \\ -n, & \text{if reticle is loaded in } n\text{-th equipment and idle} \\ n, & \text{if reticle is loaded in } n\text{-th equipment and busy} \\ -(N^s+1), & \text{if reticle is loaded at the inspection station and idle} \\ N^s+1, & \text{if reticle is loaded at the inspection station and busy} \end{cases}, i = 1, \cdots, N^t$$

$$L_k = \{l_{j,k}\}, l_{j,k} \begin{cases} 0, & \text{if lot is in the loading bay} \\ -1, & \text{if lot is in the external loading bay} \\ -(N^s+2), & \text{if lot is at the metrology station and idle} \\ (N^s+2), & \text{if lot is at the metrology station and busy} \\ -(n+1), & \text{if lot is loaded in } n\text{-th equipment and idle} \\ (n+1), & \text{if lot is loaded in } n\text{-th equipment and busy} \end{cases}, j = 1, \cdots, N^l$$

$$W_k = \{w_{z,k}\}, w_{z,k} = \begin{cases} 0, & \text{if wafer located with its lot} \\ -n, & \text{if wafer is loaded in } n\text{-th equipment and idle} \\ n, & \text{if wafer is loaded in } n\text{-th equipment and busy} \end{cases}, z = 1, \cdots, N^w$$

where N^t, N^l, N^w and N^s represent the number of reticles, lots, wafers, and stations, respectively. Additionally, let T_j be the action space at step k and the time at which lot j departs the system, $j = 1, \cdots, N^l$. Finally, let u_k, U_k, and $H_k(x_k, u_k)$ be the control action at step k, the collection of feasible actions at step k ($U_k \subseteq u_k$), and the reward for implementing the action u_k in state x_k evaluated using the base heuristic $\tilde{u}$.

3.2 Problem Formulation

The overall objective of the optimal reticle management is to dynamically and adaptively allocate reticles to the several slots available at each equipment in a way that minimizes the expected cycle time. Formally, we seek an allocation policy π^* such that, at each state $x_0, \cdots, x_T$, there exists a map of actions $\{\mu_0^*, \cdots, \mu_{T-1}^*\}$ to be taken at their respective time k, $k = 0, \cdots, T-1$, that can guide the system into sequential states that minimize the expected departure time of the final lot, $E(T^{N^l})$. This expected time is a function of the original state, x_0 and every action made during the production run, $\{u_0, \cdots, u_{T-1}\}$. Thus the objective is

$$\min \quad E(T^{N^l}) = f(x_0, u_0, \cdots, u_{T-1}) \tag{1}$$

Because the time in equation (1) is a function of every action made during the process, having the policy π^* that maps possible actions and the states that they transition to, would allow for an optimal value of $E(T^{N^l})$ to be achieved. To create this policy, we need to choose a vector of optimal control actions $\{u_0^*, \cdots, u_{T-1}^*\}$, so that the expected time at which the final

lot departs the system beginning at that x_0 is minimized. To do this, each individual optimal control action u_k^*, $k = 0, \cdots, T-1$, must be selected. At each decision step, the action u_k is to do nothing, or to swap the reticle on the tool that has completed a specific operation type, with another reticle from storage, more formally:

$$u_k = \left\{u_{n,p,s,k}\right\}_{n,p,q} \tag{2}$$

where:

$$u_{n,p,s,k} = \begin{cases} 0, & \text{if no swap is taken at step } k \text{ on equipment } n \text{ on slot } s \text{ for reticle ID } p \\ q, & \text{if at step } k \text{ reticle } p \text{ is removed from tool } n \text{ on slot } s \text{ and replaced by reticle } q \end{cases} \tag{3}$$

where, in equation (3), $q = 1, \cdots, N^t, p = 1, \cdots, N^t, s = 1, \cdots, N^c$, $q \neq p$, N^c = station reticle slots . An action in $\{u_{n,p,s,k}\}_{n,p,q}$ is active and contained in U_k only if, $r_{p,k} = -n$, and $r_{q,k} = 0$.

Hence, reticle management can be formulated as a stochastic control problem, where we seek the optimal policy for the adaptive assignment of reticles to resources/storage. In particular, the system state x_k at the k-th change requested is such that $x_{k+1} = f_k(x_k, u_k, \epsilon_k), k = 0, 1, \cdots, T-1$, where ϵ_k is the noise due to the failures that can affect the equipment (e.g., scanners, coaters, manipulation robots). We are looking to solve an optimization problem over policies π (also referred to as *closed loop control, feedback policies)*. These policies are formally a sequence of functions:

$$\pi = \left\{\mu_0, \cdots, \mu_{T-1}\right\}$$

where μ_k is a map from system states x_k into controls u_k that satisfy the control constraints (e.g., reticle availability) for all feasible states $x_k \in S_k$. Then the expected cost associated to a policy π starting in state x_0 is:

$$J_\pi(x_0) = E\left\{g_T(x_T) + \sum_{k=0}^{T-1} g_k\left(x_k, \mu_k(x_k), \epsilon_k\right)\right\}$$

where $g(\cdot)$ is an appropriate cost function. In this paper the function expresses the contribution of the reticle allocation choice at the k-th swap to the sample-path completion time. More details on the evaluation of such cost will be provided in Section 4.3.3. Since, we have a finite number of lots, we know that the dynamic program for stochastic finite horizon problems can be applied here. We have that:

Start with

$$J_T^*(x_T) = g_T(x_T)$$

Then for $k = 0, \cdots, T-1$, let

$$J_k^*(x_T) = \min_{u_k \in U_k} E\left\{g_k\left(x_k, \mu_k(x_k), \epsilon_k\right) + J_{k+1}^*\left(f_k\left(x_k, \mu_k(x_k), \epsilon_k\right)\right)\right\}. \tag{4}$$

If $u_k^* = \mu_k^*()$ minimizes the right end side of this equation, for each x_k and step k, the policy $\pi^* = \left\{\mu_0^*, \cdots, \mu_{T-1}^*\right\}$ is optimal.

This optimal policy, π^* is incredibly difficult to solve for several reasons. Chiefly, the $E(T^{N^l})$ does not have a closed form solution. This is due to the fact that each successive state is dependent on both the state before it, and the action taken at that state; decisions made in early states on can cascade down and create significant variance in later system states and the expectation $E(T^{N^l})$. Because of this, static

mathematical models and optimization techniques are difficult to apply to this problem. Instead, we propose the use of a rollout algorithm that, rather than solving for the optimal policy π^* all at once, solves for each element $\hat{u}_k^*$ sequentially.

4. Proposed Approach

Rather than solving the above optimal control problem, we approach an approximate version of this optimal allocation and solve it using the rollout algorithm (Bertsekas 2020) partnered with a digital twin. In particular, the rollout establishes a procedure that generates an action and, given a base policy for reticle management, and a digital twin, evaluates the action so that the action with the best approximate reward can be selected. The digital twin model serves as a simulated replica of a real-life process that works as a bridge between the optimizer and the real manufacturing process. The overall approach is depicted in Figure 2. Here, $H_k(x_k, u_k)$ denotes the value that the simulation of the base heuristic associates to the action u_k and $\{q, h, g\}$ are three feasible action values selected from U_k at step k.

As shown in Figure 2, a rollout algorithm requires two key components: the simulation model and the selection of controls used to achieve the result. The creation of the digital twin simulation model and base heuristic is presented in Section 4.2, and the rollout algorithm is presented in Section 4.3.

4.1 Process Overview

Figure 3 shows the conceptual model of how products and reticles flow through the system resources detailing the process summarized in Section 1.

As shown by Figure 3, the unit being processed by the resources across the several phases of the photolithography process is the *wafer*. Each wafer has attached a predetermined set of coating and scanning operations that are required to achieve the target *print* of the desired circuit, and different target circuits determine different product types and different required tooling of the scanner (i.e., different reticles are required). Wafers do not move through the system independently, rather, they are transported in *lots* with a determined size that groups wafers of the same type. When a lot reaches an equipment to receive a specific coating-scanning step, the wafers are separated from the lot and are processed sequentially. Once all wafers have completed the operation, they are returned to the lot and leave the station grouped again. Before a wafer can be printed, the appropriate reticle must be mounted on the scanner resource. In an attempt to maximize the scanner utilization, a single scanner typically holds mulitple reticles. Each reticle can only perform a specific print, henceforth can only process a specific step of a specific product type. Reticles are stored in an on-site main storage unit until they are needed. When a reticle is needed, it is removed from storage and added to the tool magazine of a photolithography machine. Overall, there are five primary locations that entities can move between. These are the *external loading bay* where lots wait before they enter the system, the *equipment loading*, the photolithography tools (equipment) containing the coater, scanner and a robotic handling system is re-

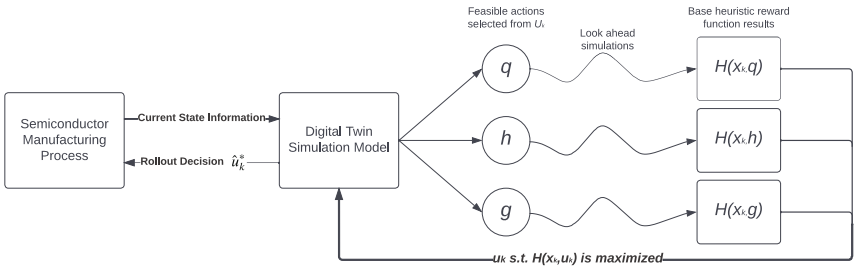


Figure 2 A Digital Twin Method for the Optimal Reticle Management

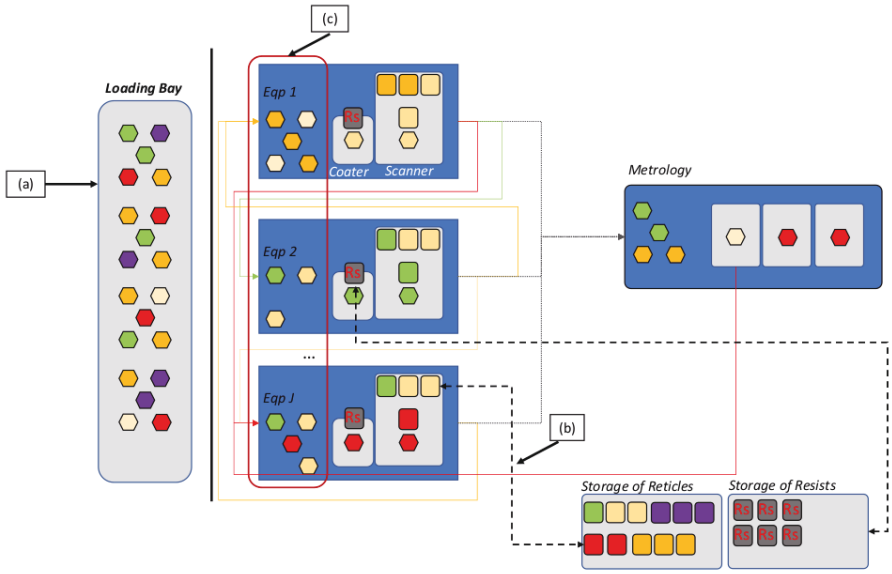


Figure 3 Semiconductor Manufacturing Process*

* Within this process behaviors function as follows (a) Lots in the loading bay are dispatched to equipment when available. This is handled by an automatic dispatcher, no decision is made here. In (b) reticles from storage are swapped with reticles on equipment to accommodate the needs of lots in the loading bay. This is the decision in the model. Finally, in (c) lots in the equipment queues are served on a FIFO bases. No decision is made here.

sponsible for moving the wafers within the equipment. The metrology station responsible to check the quality of the prints, and the inspection station responsible to check the quality of the reticles. All lots for a given simulation run are created at the beginning and added to the external loading bay. These lots are considered to be outside the system and will be prevented from entering the system when it is full. In the simulation there are a limited number of lots that can be in process at the same time. This means that the lots in the external loading bay will only enter the system when there is space; when a lot leaves the system, bringing the number of lots in the system below capacity, a new lot will enter from the external loading bay according to the predetermined order.

4.2 DTFab: the Python-based Digital Twin Model for the Photolithography

DTFab allows the user to create and characterize as instance of five major classes in the model (Figure 4 shows the entities, associ-

ated attributes and relationships in DTFab): `Reticle`, `Lot`, `Equipment`, `Product`, and `Wafer`.

The `Reticle` has the ID which is the primary key used to uniquely identify each object of this class. The operation ID is a number which matches the corresponding operation to be performed on the `Lot` or `Wafer`. The product type is a value denoting the product corresponding to the operation performed by the reticle, and the number of scans denotes the total scans performed by a reticle. In the Lot class, the Lot ID is the primary key used to uniquely identify the lot objects, and the current operation is a number which is unique for a specific layer of a wafer and matches with the operation ID of the Reticle. The metrology status attribute has boolean variables, True or False, with true denoting that the lot needs to undergo metrology while false denotes that the lot does not require metrology. Similar to the functionalities of the objects in the simulation model, the attributes are assigned to each one. Figure 4 is an Entity-Relationship model showing the classes with its individual attributes, along with their interrelationship. In figure 4, PK denotes a primary key while FK denotes a foreign key.

The model was developed in Python language version 3.88. SimPy, which is a process-based discrete-event simulation library was used as the base for our libraries. The reasons behind modeling in python, were the ease of performing optimization using reinforcement learning and the ability to create a customized simulation library for semiconductor production processes. While our current model implementation is dedicated to the photolithography process, we have designed our classes, functions, and model logic in a way that the library can be easily customized for other semiconductor processes. The Python model is also computationally efficient, which better enables its utilization as part of the digital twin system, since this requires quick decisions to be taken in real-time. In fact, initially, a commercial discrete event simulation software with detailed animation was used and while that was very helpful for ensuring correct process behavior, the speed of the software precluded using it for real-time look ahead simulations.

In the following sections, we detail the processes being modeled in DTFab.

4.2.1 Lot Operations

Lot loading and Dispatching When a lot enters the system, the lot will enter the internal loading bay. From this internal loading bay, lots are dispatched to equipment based on availability of the equipment and its ability to perform the desired operation on the wafer. In particular, when a lot is evaluated for dispatching, the system will check if an available equipment has the reticle required by the lots next operation mounted in its tool magazine. If the reticle is present on an equipment with available space in its queue, the lot is dispatched to that equipment. Otherwise, the lot is returned to the back of the queue in the internal loading bay. In the case of multiple equipment with the required reticle having space in queue, we prioritize dispatching to the system with the smallest queue. Once a lot has been dispatched to a tool, the lot enters a finite capacity queue. At the equipment level, lots are served on a FIFO basis. Once a lot has been processed, it leaves the machine to be sent to three potential locations. If the reticle used to

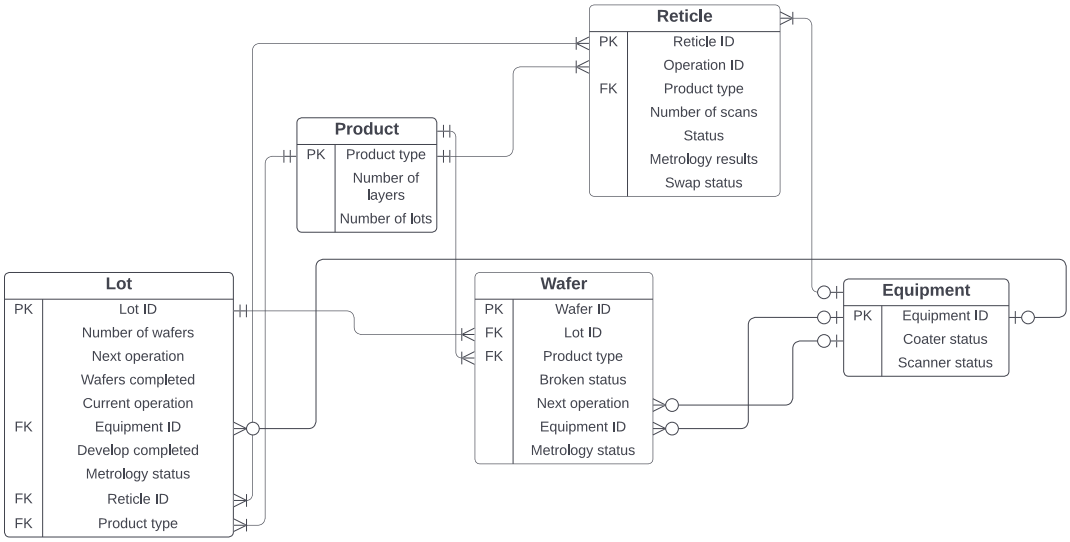


Figure 4 Entity-Relationship Model for the Classes

process the lot requires control, the lot is sent to the metrology station (more details in Section 4.2.2). Otherwise, the lot departs the system if all of its operations have been completed, or returns to the loading bay to be dispatched again if additional operations are required.

Lot Inspection After an operation, lots can be sent to metrology stations to check for potential defects. Typically, a fraction of the wafers within a lot is inspected. After inspection, the defective wafers are scrapped and the lot is sent back carrying only the healthy wafers.

4.2.2 Reticle Operations

Two main processes characterize the operating of reticles in the system: the setup, and the inspection process.

Reticle Setup The setup of reticles is performed to minimize the defects on wafers due to wrong positioning and other defects on the reticle. To ensure this, all reticle undergoes a setup when they are initially mounted. During the setup, the scanner mounting the reticle processes at lower speeds to allow for the verification of the quality of the print. The processing rate is adaptively increased until the reticle can work at regime. Due to the fact that such setup is required the first time each reticle is mounted, this impacts our allocation decision.

Reticle Inspection After each reticle has performed a predetermined number of scans, they are sent to an inspection station where the critical parameters are checked and verified. The inspection time is modeled as a stochastic process with a predetermined distribution. Post inspection, there is a probability that the reticle will still be defective and need repairing. If this occurs, after inspection, the reticle is sent to a repair station which has a constant processing time. i.e., each repair process lasts the same amount of time. After repair, the reticle is sent to the main storage to be returned to production use.

4.2.3 Description of Equipment

The tool comprises of several resources corresponding to different processes and robotic arms. Initially when lots enter an equipment,

the wafers are removed by a robotic arm and moved to a transition station. From the transition stations, the wafers are moved to baking ovens when there is available capacity to process the wafers. The baking process has a deterministic delay and a fixed capacity. A common robotic arm is used to move wafers in and out of the baking oven and coater. From the baking ovens, wafers are transferred to the coating process where the photoresist layer is added to the wafer surface. The coating machine also has a finite capacity, and the delay is deterministic. The resource has enough photoresist material to process a fixed number of wafers. When the amount of photoresist material reaches a threshold, the photoresist material is refilled to its original capacity. It is assumed that this refilling process is instantaneous in the model. Once the photoresist layer is added, the wafers are sent to a chiller station for processing. After the chiller process is completed, a robotic arm moves the wafers to the scanner resource. In the scanner resource, there are essentially two steps that are performed and each of them are modeled having a fixed capacity and deterministic processing times. The two processes are measuring and scanning. After the measure step is completed, the wafers are internally moved using a transporter. In the scanning process, UV light is projected onto the wafer surface. Between the UV light source and the Photoresist layer, a photomask or reticle is used which allows UV light to pass through and project onto the photoresist layer at certain areas. This depends on the type of circuit pattern required to be printing on the wafer surface. The scanner can mount a single reticle and can hold a fixed number of reticles in its slot at a given time. When a reticle swap is required between the mounted reticle and the slots to process incoming wafers, a robotic arm with a deterministic delay time and a fixed capacity, makes the necessary swap. This swap occurs after all the wafers in a given lot completes the scanning process. This way, the scanner is ready with the required reticle mounted to process the next lot. If the reticle required is not mounted nor available in any of the slots, a reticle swap is made between the slots and the main reticle storage. The reticle removed from the slots is the one which is not required to process any further incoming lots. This swap is done by a robotic arm with a fixed capacity and a deterministic delay. These swaps happen in parallel with the scanning process and no waiting of wafers is encountered. After scanning process is completed, a transporter moves the wafer from the scanner and a robotic arm is used to move the wafer out of the scanner. The model has two develop resources to process the wafers. The wafers move to any of the develop stations with a certain probability. Each of the develop stations has a robotic arm to transport the wafers in and out of the resource. In the develop step, the activated photoresist layer is dissolved in a developer solution and the circuit patterns are formed. Once the develop processes are completed, a robotic arm is used to group the wafers in the lot, before the lot exits the equipment.

4.3 Selecting Improved Reticle Allocations

The basic idea behind the approach is represented in Figure 5. In Figure 5, $\{q, h, g\}$ are

three feasible action values selected from U_k at step k. $H_k(x_k, u_k)$ denotes the evaluation of the reward function obtained considering the simulated sample paths generated from each action and completed using the base heuristic as a policy each time a reticle change is requested.

As mentioned in Section 3.2, the overall goal of the rollout algorithm is to solve for an approximately optimal policy $\hat{\pi}^*$. This policy consists of a series of actions $\hat{u}_k^*$, each taken to optimize an approximation of the original objective. Rather than solving for the optimal policy π^*, which is computationally prohibitive, the rollout algorithm solves for each action sequentially.

The state space for the rollout algorithm is identical to the original problem described in Section 3.2, referred to as $x_k = (R_k, L_k, W_k)$, and it is a function of the previous state, the action chosen at $k-1$, and some noise variables from the previous state ϵ_{k-1}, namely:

$$x_{k+1} = f_k\left(x_k, u_k, \epsilon_k | \epsilon_{k-1}\right), k = 0, 1, \cdots, T-1$$

where f_k is the transition function for the photolithography system (such transition can be executed through the simulation). A decision step k occurs each time a lot completes a scan operation and the equipment faces the choice of the operation to perform next. In this scenario, a reticle can potentially be changed in the equipment before the next lot begins processing. As a result of this, the time between consecutive actions is not fixed. To decide between the potential feasible actions, the rollout algorithm evaluates the approximate reward function using the DTFab simulator that implements the base heuristic as a means to manage reticle changes given an initial (state, action) pair. This estimates the performance of each action at a given state. This reward is a function of both the current state of the system k, and every action taken from k to the end of the stimulation $T-1$. That is,

$$H_k(x_k, u_k) = g(x_k, u_k, \cdots, u_{T-1} | \epsilon_k, \cdots, \epsilon_{T-1})$$

More details on the estimation of the heuristic reward are provided in Section 4.3.3.

Once the action at time k is decided, the photolithography system moves to the next decision step and the process is repeated. At the end of the simulation, the rollout algorithm will have generated a sequence of decisions: $\hat{\pi}^* = \{\hat{u}_0^*, \cdots, \hat{u}_{T-1}^*\}$. The overall procedure is summarized in Algorithm 1. In summary, based on the current state of the system, the algorithm calls a procedure for the generation of feasible actions (Section 4.3.1). For each selected action (this can be a subset of the feasible decisions) the look ahead simulations (Section 4.3.3) are performed using the base heuristic (Section 4.3.2) to evaluate the effect of the action.

Algorithm 1 Rollout Algorithm

Step 1: Create current action space U_k from feasible actions

Step 2: For each potential action in U_k, run a look ahead function using the base heuristic $\tilde{u}$

Step 3: Select the control that maximizes the reward function $H_k(x_k, u_k)$ as $\hat{u}_k^*$ and execute this action (set $u_k = \hat{u}_k^*$)

Step 4: Update the state information and move to x_{k+1}

Step 5: Repeat steps 1-4 until the simulation has reached its terminating condition

Step 6: Return $\hat{\pi}^* = \{\hat{u}_0^*, \cdots, \hat{u}_{T-1}^*\}$

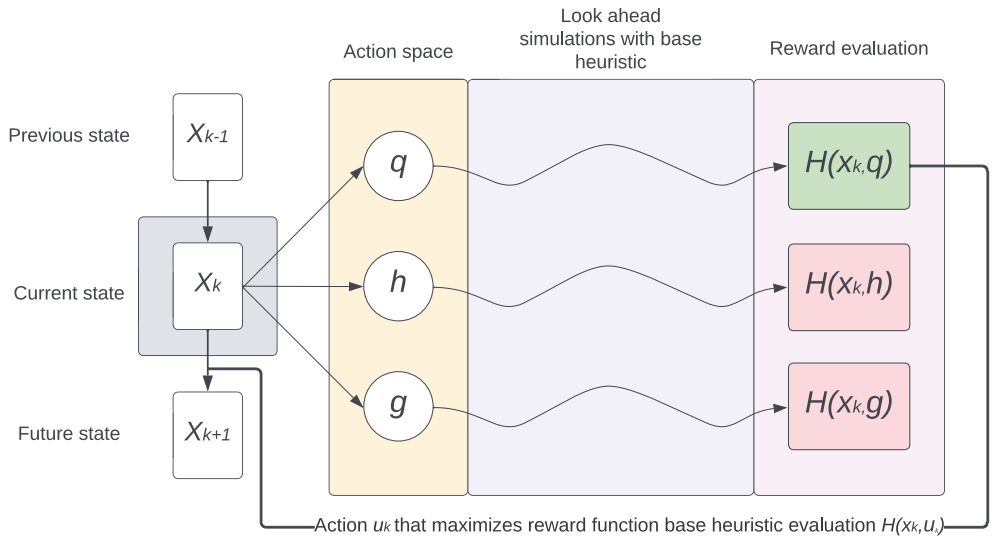


Figure 5 Rollout Algorithm

4.3.1 Defining the Action Space

The action space U_k contains all the reticle allocations that are feasible at decision step k. In other words, at step k the system compiles a list of every reticle switch that is feasible to the current state. Each of these switches is evaluated for potential, and a winner is chosen to be executed based on its performance in the look ahead function discussed in Section 4.3.3. Before this can be done however, the system must first create U_k from the current state information To create the action space, the first step the algorithm must perform is determining which reticles are feasible to switch. Given that the decision occurs at station n at step k. The system defines two subsets of R_k, O_k^n and O_k^u. Define O_k^n ($O_k^n \subseteq R_k$) as the set of reticles needed by lots in the loading bay at step k, that is any reticle whose operation number attribute matches the next operation of any lot in the loading bay or the queue of station n. Additionally, define O_k^u ($O_k^u \subseteq R_k$) as the set of reticles not needed by lots in the loading bay at step k, that is any reticle whose operation number attribute does not match the next operation of any lot in the loading bay. In set notation, this selection can be defined as $U_k = \{\{u_{n,p,s,k}\}_{n,p,q} : r_{p,k} = -n, r_{q,k} = 0, r_{p,k} \in O_k^u, r_{q,k} \in O_k^N\}$

The algorithm process is shown in algorithm 2.

Algorithm 2 Creating the Action Space U_k

Step 1: Define the equipment where the decision is being made as index n

Step 2: Add any actions fulfilling $r_{p,k} = -n, r_{q,k} = 0, r_{p,k} \in O_k^u, r_{q,k} \in O_k^N$ to the feasible action space U_k

Step 3: Return the action space U_k

4.3.2 Reticle Management Base Heuristic

In the rollout algorithm, a base heuristic is needed to control the decisions made in the model within the look ahead simulations. In the algorithm, the base heuristic $\tilde{u}$ functions by marking reticles in storage for a future swap.

When a lot in *equipment loading* fails to find the needed reticle on an available equipment, one reticle matching the lots currently needed in storage is marked. This is done by changing the reticle's Swap status attribute to **TRUE**. Let R_k^m be the subset of reticles in storage that is currently marked, $R_k^m \subseteq R_k$. When a decision is made using $\tilde{u}$, the heuristic selects one reticle the subset of marked reticles, and the reticle from the equipment's slots whose operation number is required the least by the next operation of lots in the loading bay and queue. Demand ties are broken by lowest slot index c. If we define the location of the decision as equipment n and the decision of the heuristic at step k of the look ahead $\tilde{u}_k$. The procedure is reported in Algorithm 3.

Algorithm 3 Base Heuristic Procedure

Step 1: Define the equipment where the decision is being made as index n

Step 2: Choose a feasible reticle from slots of n to return

Step 3: Choose a reticle from R_k^m to be mounted on equipment n. Change this reticle's Swap status to **FALSE**

Step 4: Set this action as $\tilde{u}_k$

4.3.3 The Lookahead Evaluation

This step is necessary to choose an action from the feasible set U_k. In particular, each element of the set is evaluated using the look ahead simulation. A number of replications are performed, and for each replication, only the first reticle allocation is decided through the action u_k, while all the forthcoming reticles are allocated using the base policy. Once the simulations have completed, the evaluation for each action is made available as the base heuristic reward $H_k(x_k, u_k)$, and the action u_k is chosen such that $H_k(x_k, u_k)$ is maximized. In our specific application, each action is evaluated in terms of the completion time of the lots currently present in the system.

More specifically, at each decision point (i.e., reticle change request), the algorithm evaluates all the actions $u_k \in U_k$. The reward $H_k(x_k, u_k)$ is then evaluated using the rollout mechanism: look ahead simulations are initialized with the current system state x_k and the action to be evaluated $u_k \in U_k$. Within each lookahead simulation, when a reticle change is requested the base heuristic is used to determine the decision (Section 4.3.2). The simulations result in a number of trajectories n_{LA} for the future states, which we use to estimate the expected reward for each action. It is important to highlight that these replications are run in parallel within our architecture. The lookahead simulations run until a terminating condition is reached. In this work, we propose two alternative terminating conditions that result into two rollout policies.

Rollout Policy 1. The first implementation has the simulations terminate when all the lots present in the system at the time when the reticle change is called for, are completed. We refer to this completion time as $T_r^{N^l}$, with $r = 1, \cdots, n_{LA}$. In this case, the reward is $H_k = -E\left(T^{N^l}\right)$, which we estimate through n_{LA} replications.

Rollout Policy 2. The second implementation has the simulations terminate when all the lots present in the system at the time when the reticle change is called for, complete their next operation. We refer to this completion time as $O_r^{N^l}$, with $r = 1, \cdots, n_{LA}$. In this case

the reward $H_k = -E\left(O^{N^l}\right)$, which we estimate through n_{LA} replications. It is apparent that the second policy uses myopic simulations only looking at the next operation, thus resulting in potential computational savings (the two policies become increasingly similar as the state gets closer to the completion of the lots operations).

Once all of the look ahead simulations have concluded (irrespective of rollout policy chosen), the actions are compared based on the base heuristic evaluation $H_k(x_k, u_k)$, and the action u_k maximizing $H_k(x_k, u_k)$ is selected as $\hat{u}_k^*$. This process is summarized in Algorithm 4.

Algorithm 4 The Look Ahead Function

Step 1: For each possible action in U_k, create a look ahead simulation.

Step 2: Run look ahead simulations to completion

Step 3: Return action u_k from U_k such that the base heuristic evaluation, $H_k(x_k, u_k)$, is maximized

Step 4: Set this action as $\hat{u}_k^*$

4.4 Contribution

As highlighted from the literature analysis, most of the approaches do not leverage detailed simulations of the system, and rather focus on building approximations of the reward through surrogates. This is not aligned with the increasing availability of digital copies of complex systems and processes, and the plethora of information that can be gathered from resources in real time. On the other hand, heuristic policies developed by industry experts should be accounted for when generating potentially new approaches. However, the approaches in the literature do not have a way to embed policies to guide the search.

In this manuscript, we focus on the problem of improving cycle times within the photolothography process in a semiconductor fab with the goal to show that simulation driven reinforcement learning can lead to performance improvement over the policies adopted in the industry practice. Due to the fact that the loss function is associated with the cycle time of the lots in the system, we have no closed form reward (similar to other Reinforcement Learning approaches in the literature). This prevents us from using exact stochastic dynamic programming. Considering approximate dynamic programming, we could have designed a surrogate for the cost function, i.e., a metamodel of the cycle time as a function of the reticle choice. Nonetheless, the categorical nature of the action space does not make it amenable to most learning models. Moreover, embedding base policies of practical relevance that exist for the reticle management problem is desired. To this end, a rollout Bertsekas (2020) based framework for reticle management is developed that: (i) allows us to solve the control problem without a analytical knowledge of the cost function; (ii) can be applied to categorical decisions; (iii) has a base heuristic as an immediate mechanism to embed pre-existing policies and improve them. In light of these observations, the contribution of this paper is two-fold:

Contribution 1. We develop a Python simulation that represents the digital copy of the system under analysis. This digital copy can take as input the state of the system at any point in time and simulate any reticle allocation action by warm-starting a discrete event

dynamical simulation of the photolithography system. This allows, in principle, to estimate multiple metrics of the system which we may be interested in optimizing. The simulator is in the DTFab package which will be made available with the published manuscript.

Contribution 2. We design a rollout algorithm that can intelligently generate reticle allocations, while using the simulation to evaluate them. Different from the approaches in the literature, we can directly use the simulator to evaluate the performance of the system, without estimating any surrogate of the response. This approach eliminates the need to train a surrogate model at the cost of requiring few online simulations to evaluate the policy. Hence, it is important to develop a computationally efficient simulator.

5. Experimental Results

In this section, we present the main empirical results shown for the proposed approach. In particular, our experiments focus on two main objectives: DTFab is validated statistically in Section 5.1. We defined a set of experiments with our industry partner with the objective to evaluate the accuracy. The rollout-based approach is verified in Section 5.2, by evaluating the performance gain introduced by the policy in terms of average lot cycle time reduction. The rollout algorithm and the results will be publicly available in the form of a GitHub report upon acceptance of the manuscript.

5.1 Model Validation

In collaboration with our partners at Intel, we designed a test benchmark to statistically demonstrate the validity of the simulator. In particular, we evaluated the impact of two main factors: (i) number of copies for each reticle (i.e., reticles that can perform the same operation); (ii) Lot release policies (i.e., the way lots of different product types are loaded in the external loading bay). For reticle copies, we considered two different levels, running the system with two and three copies, respectively. For lot release policies, we consider two possible cases, batch release and cascade release: (i) in the batch release case, lots enter the system in homogeneous groups of a fixed size; (ii) in the cascade case, lots enter the system sequenced at random. Table 1 reports the key input parameters and associated levels.

5.1.1 System Metrics Formulation

We collected several performance metrics to characterize the system level performance. These performance metrics are collected in experiments replicated 30 times, that is $N^r = 30$. Let r represent the r-th replication, $r = (1, \cdots, N^r)$. These metrics are, the cycle time of each of the N^l lots $LCT_{j,r}$, $j = (1, \cdots, N^l)$, the utilization of both the coater and scanner on each of the N^s stations, which we refer to as $UTC_{n,w}$, with $n = (1, \cdots, N^s)$ and $UTS_{n,w}$, $n = (1, \cdots, N^s)$. In the following, we provide the detailed calculation for each metric and the obtained results.

$$LCT_{j,r} = T_{j,r} - E_{j,r}, j = (1, \cdots, N^l)$$

where $T_{j,r}$ and $E_{j,r}$ denote the time where lot j departs and enters the system in replication r, respectively.

$$UTC_{n,r} = \frac{\sum_{j=1}^{N^l} \sum_{i=1}^{\infty} f^c_{n,j,i,r} - s^c_{n,j,i,r}}{T_r^{N^l}}$$

Where $f^c_{n,j,i,r}$ and $s^c_{n,j,i,r}$ denote the time at which lot j respectively finishes and starts pro-

Table 1 Model Input Parameters

Parameter Type	Variable	Distribution	Value	Units	Description
Simulation run parameters	N^l	N/A	750	-	Lots completed in simulation
	N^w	N/A	30	-	Number of replications
Tool parameters	N^c	N/A	4	-	Number of reticle storage slots
	N_s^m	N/A	1	-	Number of scanners (each mounting a reticle)
Equipment parameters	N_s	N/A	15	-	Total number of equipment
	N/A	Uniform	0.5 to 6	Days	Time between failure for coater/scanner
	N/A	Triangular	1 to 6	Hours	Repair time
	N^c	N/A	5	-	Lot capacity per equipment
Product parameters	N^p	N/A	30	-	Number of product types
	N^o	N/A	30	-	Number of layers per product
	N/A	Uniform	43.00	Seconds	Coating time
	N/A	N/A	25	Seconds	Scanner time
Lot parameters	N^f	N/A	25	-	Number of wafers per lot
Reticle parameters	N/A	N/A	1350	-	Total number of operations across product types
	N^t	N/A	4050	-	Total number of reticles in system (3 copy case)
	N^t	N/A	2700	-	Total number of reticles in system (2 copy case)
	N/A	N/A	75	-	Total number of reticles allocated for use by all tools
	N/A	N/A	500	-	Max number of scans per reticle before inspection
	N/A	N/A	42	Seconds	Reticle swap time for storage to slot
	N/A	N/A	21	Seconds	Reticle swap time for slot to mount
Resist parameters	N/A	N/A	300	-	Number of coating cycles before refill
Inspection parameters	N/A	Triangular	5 to 15	Minutes	Reticle inspection time
	N/A	Triangular	2 to 3	Hours	Lot inspection time in metrology
Queuing policy	N/A	N/A	N/A	-	FIFO

cessing on the coater of station n for the ith time in replication r.

$$UTS_{n,r} = \frac{\sum_{j=1}^{N^l} \sum_{i=1}^{\infty} f^s_{n,j,i,r} - s^s_{n,j,i,r}}{T_r^{N^l}}$$

where $f^s_{n,j,i,r}$ and $s^s_{n,j,i,r}$ denote the time at which lot j respectively finishes and starts processing on the scanner of station n for the ith time in replication r.

To evaluate the performance of the overall system, these metrics were then averaged into system wide metrics using the following formulations. The sample mean of each metric is denoted by the bar notation. That is

$$\overline{LCT}_r = \sum_{j=0}^{N^l} \frac{LCT_{j,r}}{N^l}$$

$$\overline{UTC}_r = \sum_{n=0}^{N^s} \frac{UTC_{n,r}}{N^s}$$

$$\overline{UTS}_r = \sum_{n=0}^{N^s} \frac{UTS_{n,r}}{N^s}$$

For each replication r we also collect the average per reticle swaps made between main storage $\overline{SWS}_r$, the average per reticle swaps made within equipment's slots $\overline{SWE}_r$, and the average scans per reticle $\overline{ASC}_r$. That is

$$\overline{SWS}_r = \sum_{n=0}^{N^s} \frac{Y_{n,r}}{N^t}$$

$$\overline{SWE}_r = \sum_{n=0}^{N^s} \frac{Z_{n,r}}{N^t}$$

$$\overline{ASC}_r = \frac{C_r}{N^t}$$

where $Y_{n,r}$ represents the number of swaps made between equipment n and storage; $Z_{n,r}$ represents the number of times a reticle was moved to equipment n's reticle mount from equipment slots; and C_r represents the total number of scans performed in replication r. The average quantity of wafers inspected, the average number of wafers rejected by metrology, and the average proportion of defective wafers are collected per replication as well, and these will be referred to as $\overline{QTI}_r$, $\overline{WRM}_r$, and $\overline{PDF}_r$, respectively.

$$\overline{QTI}_r = \sum_{k=0}^{N^w} \frac{U_{k,r}}{N^w}$$

$$\overline{WRM}_r = \sum_{k=0}^{N^w} \frac{V_{k,r}}{N^w}$$

$$\overline{PDF}_r = \sum_{k=0}^{N^w} \frac{W_{k,r}}{N^w}$$

We then derive the averaged measures, and associated confidence interval, across independent macro-replications, namely:

$$\overline{\overline{LCT}} = \sum_{r=0}^{N^r} \frac{\overline{LCT}_r}{N^r}, \overline{\overline{UTC}} = \sum_{r=0}^{N^r} \frac{\overline{UTC}_r}{N^r}$$

$$\overline{\overline{UTS}} = \sum_{r=0}^{N^r} \frac{\overline{UTS}_r}{N^r}, \overline{\overline{SWS}} = \sum_{r=0}^{N^r} \frac{\overline{SWS}_r}{N^r}$$

$$\overline{\overline{SWE}} = \sum_{r=0}^{N^r} \frac{\overline{SWE}_r}{N^r}, \overline{\overline{ASC}} = \sum_{r=0}^{N^r} \frac{\overline{ASC}_r}{N^r}$$

$$\overline{\overline{QTI}} = \sum_{r=0}^{N^r} \frac{\overline{QTI}_r}{N^r}, \overline{\overline{WRM}} = \sum_{r=0}^{N^r} \frac{\overline{WRM}_r}{N^r}$$

$$\overline{\overline{PDF}} = \sum_{r=0}^{N^r} \frac{\overline{PDF}_r}{N^r}$$

5.1.2 Analysis

Table 2 shows the cycle time metrics across the tested conditions, while Figure 6b reports box plots giving information on the distribution of the metrics. From the results, we can observe that the average lot cycle times for the two copies case is less than the three copies case for both cascade and batch dispatch scenarios. This can be explained by the fact that in the three copies case, a greater number of reticles

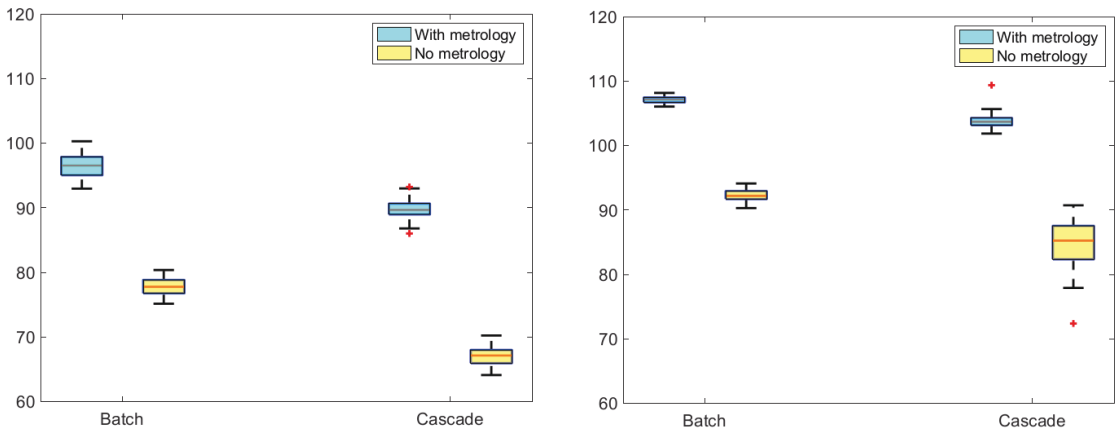
are required to undergo setup in the system. This means that more number of lots are required to be sent to metrology to setup these reticles. Figure 7a and Table 2 also show the average cycle times of the lots that skip metrology, which means that these lots are not used to setup any reticles. This gives us an insight on the absolute difference between the two lot dispatch scenarios. From the above table, we find that among the two reticle cases, the average lot cycle times are much lower for a two copies case. The reason behind this is that in the two copies case, there are a total of 2700 reticles which complete setup much faster compared to the 4050 reticles in the three copies case. We find that even after removing the lots that go to metrology for setup, we find a difference between batch and cascade scenarios where the average lot cycle time for the cascade scenario is much lesser compared to the batch for both reticle cases. In the case of a batch dispatch scenario, each product type arrives in batches of size five. This means that during the initial phase of the simulation, where the number of reticles setup are less, more lots compete for the same reticles. From Section 5, we know that all reticles require setup. Due to this, in the case of the batch scenario, the lots have a higher waiting time in the system before a reticle is setup and is available for process. In the case of cascade, the lot product types are evenly spaced out as they are assigned at random among the 750 total lots. Since the lot product types are evenly spaced out, there is less competition among lots to use specific reticles for process. In cascade, as further lots of the same product types enter the system, it encounters reticles that are setup for process which reduces the wait time of lots.

Table 3 shows three metrics pertaining to reticle swap, the average number of swaps between the main storage and the equipment slots, the average number of swaps within the equipment and the average number of scans per reticle. The average value for the first metric is calculated by dividing the total number of reticle swaps done between the reticle and the main storage, with the total number of reticles in system. A two copies reticle case has 2700 reticles in system, while the three copies case has 4050 reticles in system. The other metrics are calculated in a similar fashion. The results show that the number of reticle movements between the main storage and the equipment slots are higher for cascade by 65% in the 2 reticle copies case, and this percentage decreases to 42.8% in the three reticle copies case. This can be accounted by the diversity in product types in the case of cascade, which would require a higher number of reticle swaps from storage. This also explains the results obtained for the average number of reticle swaps from the waiting slots to begin operating on the wafers.

Table 4 shows the performance metrics obtained for the equipment. Figure 7b provides the same results graphically to show the distribution of the metric across independent simulation replications. From the results, we can observe that the coater utilization is not statistically impacted by the number of copies per reticle. Furthermore, we notice that the coater utilization is higher than the scanner across all experiment conditions. As discussed in Section 4, the busy time for the scanning process in the tool is contributed by only two steps,

Table 2 Average Lot Cycle Time and Standard Error when Including All Lots, and Removing Lots That Are Sent to Metrology

Reticle case	Release policy	$\overline{\overline{CT}} \pm$ SE (with metrology)	$\overline{\overline{CT}} \pm$ SE (no metrology)
2 copies	Batch	96.563 ± 0.330	77.886 ± 0.249
	Cascade	89.683 ± 0.304	67.213 ± 0.285
3 copies	Batch	107.098 ± 0.096	92.259 ± 0.158
	Cascade	103.768 ± 0.246	84.411 ± 0.753



(a) Case with two reticle copies

(b) Case with three reticle copies

Figure 6 Average Lot Cycle Times Analysis

while all other processes contribute to the busy time of the coater. This reason contributes to the higher utilizations for coater as compared to the scanner in all cases.

Finally, Table 5 shows the proportion defective of wafers inspected in the metrology station. We perform this study to verify the correctness of the wafer inspection process. Each wafer that is inspected has the same probability to fail an inspection according to the model. This is reflected by the results showing that the proportion defective value is statistically independent from the condition.

5.2 Rollout Algorithm Experiments

To evaluate the impact of the novel reticle management framework, we study the impact of the rollout-based policy when applied to a photolithography system looking at the effect of the number of the number of lots in the system, the total number of machines, the total number of layers per wafer (number of operations), hence the number of reticles in the system. In addition to the two rollout policies introduced in Section 4.3.3, we consider the following state of the art production control rules:

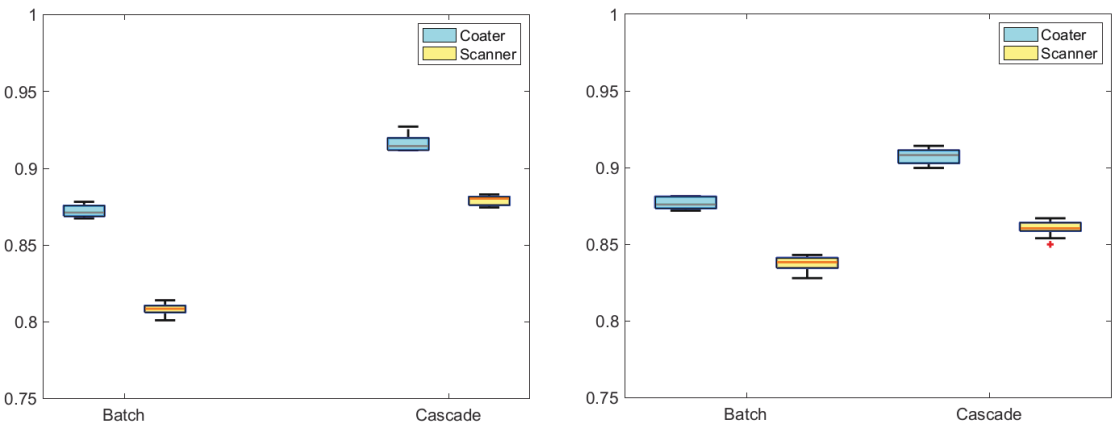
- Shortest Processing Time (SPT): when the swap is called for, if available, the reticle associated to the lots requesting the shortest cumulative processing time is mounted;
- Longest Processing Time (LPT): when the swap is called for, if available, the reticle associated to the lots requesting the longest cumulative processing time is mounted.

Table 3 Reticle Swap Results across Experimental Conditions and Standard Error (se) across 30 Replications

Reticle case	Release policy	$\overline{\overline{\text{SWS}}} \pm \text{SE}$	$\overline{\overline{\text{SWE}}} \pm \text{SE}$	$\overline{\overline{\text{ASC}}} \pm \text{SE}$
2 copies	Batch	7.880 ± 0.279	9.399 ± 0.0584	324.921 ± 0.829
	Cascade	13.078 ± 0.0434	12.420 ± 0.0329	326.300 ± 0.7309
3 copies	Batch	6.175 ± 0.0376	6.329 ± 0.03169	216.702 ± 0.293
	Cascade	8.818 ± 0.067	8.250 ± 0.0319	217.745 ± 0.286

Table 4 Equipment Utilization Results across Experimental Conditions and Standard Error (se) across 30 Replications

Reticle case	Release policy	$\overline{\overline{\text{UTC}}} \pm \text{SE}$	$\overline{\overline{\text{UTS}}} \pm \text{SE}$
2 copies	Batch	0.872 ± 0.00193	0.808 ± 0.002083
	Cascade	0.916 ± 0.00282	0.879 ± 0.00153
3 copies	Batch	0.877 ± 0.0187	0.837 ± 0.00168
	Cascade	0.907 ± 0.00248	0.86 ± 0.00164



(a) Case with 2 reticle copies

(b) Case with 3 reticle copies

Figure 7 Equipment Utilization Analysis

Table 5 Wafer Inspection Results across Experimental Conditions and Standard Error (se) across 30 Replications

Reticle case	Release policy	$\overline{\overline{\text{QTI}}} \pm \text{SE}$	$\overline{\overline{\text{WRM}}} \pm \text{SE}$	$\overline{\overline{\text{PDF}}} \pm \text{SE}$
2 copies	Batch	234696.8 ± 1070.636	154.60 ± 4.54	0.000658 ± 0.00424
	Cascade	182366.2 ± 1846.128	122.1 ± 4.094	0.000669 ± 0.00221
3 copies	Batch	327398.4 ± 1560.437	218 ± 6.375	0.000665 ± 0.00408
	Cascade	258794.9 ± 3087.738	172.8 ± 5.968	0.000667 ± 0.00193

For the purpose of evaluating the quality of the different policies, we collect five different output metrics: the average lot cycle time (in hours), equipment utilization for the scanner and coater, the number of retciles swapped between tools and storage, and the number of reticle exchanged within equipment. For the formulas used in the calculations of these metrics, please see Section 5.1.1. The resulting design of experiment factors and levels are provided in Table 6. We used $n_{LA} = 15$ simulation replications to evaluate the action effect online for the rollout policies, and macro-replicated all the experiments 15 times to calculate the confidence intervals over the output metrics.

In the following, we discuss the results separately for the different output metrics.

Cycle time. We start analyzing the effect of the policies on the cycle times which are optimized by the rollout. The results are in Tables 7-8 (first column) for the small and larger system, respectively, and Figures 8a-8b for the small and large system respectively, the average cycle times of the lots for the Base Heuristic, SPT and LPT policies are similar with no statistical difference considering a significance $\alpha = 0.05$, for both the smaller and larger systems. We observe that, compared with the Base Heuristic, the Rollout1 policy results in an average cycle time reduction of 6.93% for the smaller system which increases to 13.45% for the larger system. The Rollout2 results in the average lot cycle time reducing by 7.30% for the smaller system and by 13.30% for the larger system, when compared with the Base Heuristic. We can see that across all cases the rollout methods are statistically superior to the alternative policies, with a computational advantage for Rollout2 policy as discussed in Section 4.3.3.

Equipment Utilization. The equipment utilization in Tables 7-8 is separately calculated for the coater $\left(\overline{\overline{UTC}}\right)$ and the scanner $\left(\overline{\overline{UTS}}\right)$. Figures 9a-9b show the performance for the scanner in yellow and in blue for the coater. We observe that the average scanner and coater utilization for the Base Heuristic, SPT and LPT policies are similar with no statistical difference with a significance $\alpha = 0.05$, for both the smaller and larger systems. We also see that Rollout1 policy results in the average scanner utilization increasing by 3.49% for the smaller system and by 21.35% for the larger system, when comparing with the Base Heuristic. The Rollout2 policy results in the average scanner utilization increasing by 13.37% for the smaller system and by 19.66% for the larger system, when comparing with the Base Heuristic. Similarly, when using the Rollout1 policy, the average coater utilization increased by 4.15% for the smaller system and by 23.47% for the larger system. The Rollout2 policy results in the average coater utilization increasing by 35.84% for the smaller system and by 21.51% for the larger system.

Reticle Swaps. The reticle swaps are separately reported in Tables 7-8 for the swaps with storage $\left(\overline{\overline{SWS}}\right)$ and the and within equipment $\left(\overline{\overline{SWE}}\right)$. Figures 10a-10b show the swaps with the storage in blue and within the equipment in yellow. We observe that the average swaps between main storage and slots remains almost the same while the average swaps within the equipment shows a statistically significant increase for both the Rollout policies. This means that to improve cycle time and utilization result in increased movements of reticles within the

Table 6 Rollout Experiments Parametrization*

Parameter Type	Variable	Distribution	Value	Units	Description
Simulation run parameters	N^l	N/A	[10, **30**]	-	Lots completed in simulation
	N^w	N/A	15	-	Number of replications
Tool parameters	N^c	N/A	2	-	Number of reticle storage slots
Rollout parameter	n_{LA}	N/A	15	-	Number of replications for rollout
Equipment parameters	N_s	N/A	[2, **10**]	-	Total number of equipment
Product parameters	N^p	N/A	3	-	Number of product types
	N^o	N/A	[3, **5**]	-	Number of layers per product
Reticle parameters	N/A	N/A	[9, **15**]	-	Total number of operations across product types
	N^t	N/A	[27, **45**]	-	Total number of reticles in system (3 copy case)
	N/A	N/A	[6, **30**]	-	Total number of reticles allocated for use by all tools

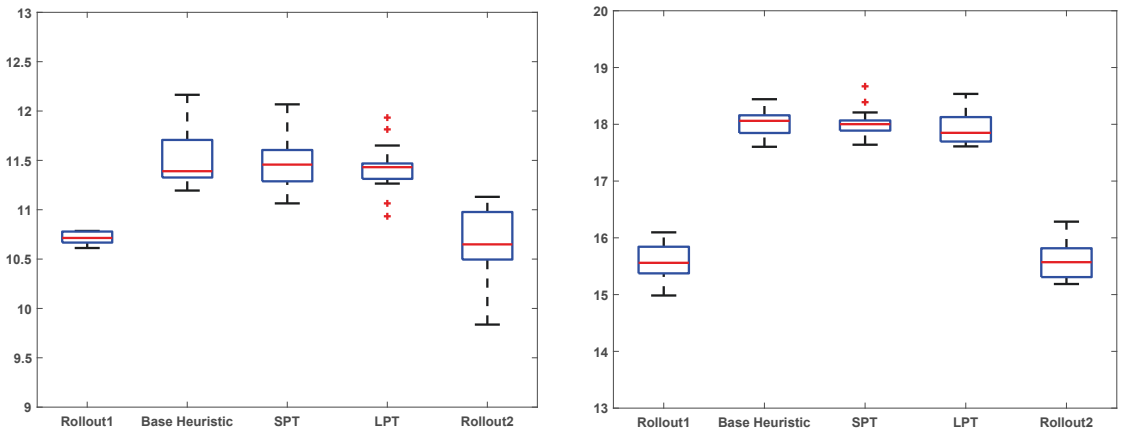
*The parameters for the larger system are provided in bold, where no bold value is supplied the experiments were run for the smaller system.

Table 7 Results for Smaller System across 15 Replications

Policy	$\overline{\overline{\text{CT}}}$ ± SE	$\overline{\overline{\text{UTC}}}$ ± SE	$\overline{\overline{\text{UTS}}}$ ± SE	$\overline{\overline{\text{SWS}}}$ ± SE	$\overline{\overline{\text{SWE}}}$ ± SE
Base Heuristic	11.514 ± 0.07718	0.505 ± 0.00531	0.441 ± 0.00349	0.703 ± 0.00722	0.837 ± 0.00484
SPT	11.455 ± 0.06245	0.492 ± 0.00746	0.433 ± 0.00394	0.703 ± 0.00626	0.837 ± 0.00704
LPT	11.423 ± 0.06505	0.495 ± 0.00734	0.435 ± 0.00405	0.698 ± 0.00710	0.832 ± 0.00493
Rollout 1	**10.7157 ± 0.015**	0.526 ± 0.00397	0.4564 ± 0.00544	**0.7596 ± 0.00501**	0.8467 ± 0.00518
Rollout 2	**10.673 ± 0.09316**	**0.686 ± 0.01138**	**0.500 ± 0.00779**	**0.756 ± 0.00674**	0.857 ± 0.00720

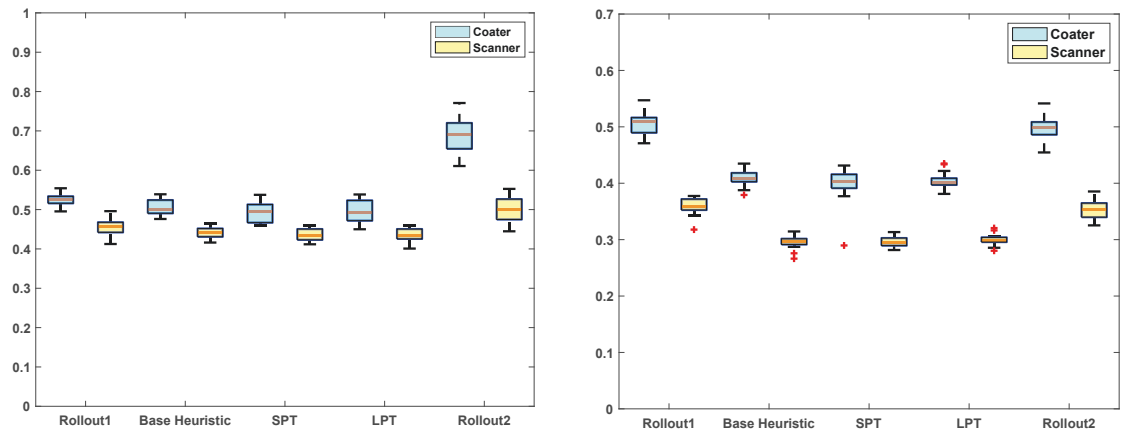
Table 8 Results for Larger System across 15 Replications

Policy	$\overline{\overline{\text{CT}}}$ ± SE	$\overline{\overline{\text{UTC}}}$ ± SE	$\overline{\overline{\text{UTS}}}$ ± SE	$\overline{\overline{\text{SWS}}}$ ± SE	$\overline{\overline{\text{SWE}}}$ ± SE
Base Heuristic	18.013 ± 0.06807	0.409 ± 0.00390	0.295 ± 0.00318	0.547 ± 0.00675	1.395 ± 0.01338
SPT	18.021 ± 0.06785	0.398 ± 0.00879	0.296 ± 0.00245	0.538 ± 0.01095	1.371 ± 0.01510
LPT	17.925 ± 0.07109	0.404 ± 0.00386	0.299 ± 0.00263	0.549 ± 0.00664	1.377 ± 0.01812
Rollout 1	**15.589 ± 0.07492**	**0.505 ± 0.00581**	**0.358 ± 0.00403**	**2.035 ± 0.03871**	**1.759 ± 0.02306**
Rollout 2	**15.616 ± 0.09378**	**0.497 ± 0.00612**	**0.353 ± 0.00421**	**1.955 ± 0.04821**	**1.638 ± 0.01644**



(a) Small System (b) Large System

Figure 8 Cycle Time Performance Across Competing Policies



(a) Small System (b) Large System

Figure 9 Scanner Utilization Across Competing Policies

system. Figures 10a-10b show that the average swaps between main storage and slots, and the average swaps within the equipment increases substantially for both the Rollout policies in the larger system. The bold results in Tables 7-8 are statistically significant at $\alpha = 0.05$, signifying that the increased movement of reticles resulting from the Rollout policies is a significant effect of the introduced control.

6. Conclusion and Future Work

In this paper, we presented a digital twin-driven rollout method for the reticle management problem for photolithography in chip manufacturing. To this end, a novel digital twin discrete-event simulation model of the photolithography process was developed and validated. We analyzed and statistically validated the simulation model designing a set of experimental conditions that varied the number of copies of each reticle type, and the lot release policy. We observed statistics on lot cycle times, resource utilization, reticles movements and verified the robustness of the results for the different conditions. These experiments were run considering a heuristic rule for reti-

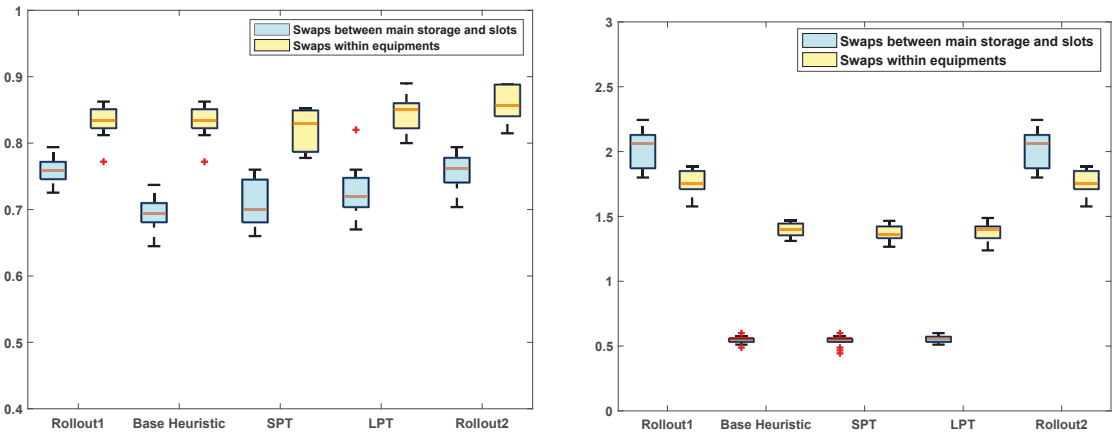


Figure 10 Reticle Swaps Across Competing Policies

cles management that was inspired to the real system thanks to the collaboration with the co-authors at Intel. Given the verified simulator, we proceeded to apply the rollout and evaluated the empirical effect of the new policy. We observed a significant improvement in the average lot cycle time when the rollout algorithm was utilized over the base heuristic policy. Although the approach presented in this paper proved to be successful for a scaled model, it exposed the need for further work to be done to scale the algorithmic platform. Additionally, the implementation of the Python digital twin model has been designed with future growth in mind. Extending the current implementation of the Photolithography process to encompass other processes is another major step currently under way.

Acknowledgments

The authors thank the reviewers for the insightful comments that helped improving the manuscript. This work was partially supported by the Intel Research under Grant No.00035705, and the NSF-CISE under Grant No.2000792.

Data Availability

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

References

Allgeier H, Flechsig C, Lohmer J, Lasch R, Schneider G, Zettler B (2020). Simulation-based evaluation of lot release policies in a power semiconductor facility - A case study. *2020 Winter Simulation Conference (WSC)*. USA.

Benzoni A, Yugma C, Bect P, Planchais A (2020). Allocating reticles in an automated stocker for semiconductor manufacturing facility. *2020 Winter Simulation Conference (WSC)*.USA.

Bertsekas D (2020). *Rollout, Policy Iteration, and Distributed Reinforcement Learning*. Athena Scientific, Belmont, Massachusetts, USA.

Bixby R, Burda R, Miller D (2006). Short-interval detailed production scheduling in 300mm semiconductor manufacturing using mixed integer and constraint programming. *The 17th Annual SEMI/IEEE ASMC 2006 Conference*. USA.

Bradley S, Hax A, Magnanti T (1977). *Applied Mathematical Programming*. Addison-Wesley, USA.

Byrne P (2007). An analysis of semiconductor reticle management using discrete event simulation. *Proceedings of the 2007 Summer Computer Simulation Conference*. USA.

Carranza R (1986). Silicon Run LITE, an overview film that introduces semiconductor manufacturing.

NationalElectrical.https://siliconrun.com/our-films/silicon-run-lite/.

Dantzig G, Thapa M (1997). *Linear Programming*. Springer, USA.

Derbyshire K (2015). EUV: Cost killer or savior? *Semiconductor Engineering*. https://semiengineering.com/euv-cost-killer-or-cost-savior/.

De Diaz S, Fowler J, Pfund M, Mackulak G, Hickie M (2005). Evaluating the impacts of reticle requirements in semiconductor wafer fabrication. *IEEE Transactions on Semiconductor Manufacturing* 18: 622-632.

EESEMI (2005). Mask and ceticles: Tools for pattern formation on semiconductor wafers. *EESEMI*. https://eesemi.com/masks-reticles.html.

Fathi Y, Barnette K (2002). Heuristic procedures for the parallel machine problem with tool switches. *International Journal Of Production Research* 40: 151-164.

Fox M (1983). Constraint-directed search: A case study of Job-Shop scheduling. PhD thesis, Carnegie-Mellon University.

Hadash G, Kermany E, Carmeli B, Lavi O, Kour G, Jacovi A (2018). Estimate and replace: A novel approach to integrating deep neural networks with existing applications. *ArXiv Preprint ArXiv*:1804.09028.

Ham A, Cho M (2015). A practical two-phase approach to scheduling of photolithography production. *IEEE Transactions on Semiconductor Manufacturing* 28: 367-373.

Haße H, Li B, Weißenberg N, Cirullies J, Otto B (2019). Digital twin for real-time data processing in logistics. *Artificial Intelligence and Digital Transformation in Supply Chain Management: Innovative Approaches for Supply Chains. Proceedings of the Hamburg International Conference of Logistics (HICL)* 27: 4-28.

Hickie M (1999). Improving photolithography reticle management with network modeling and discrete event simulation. PhD thesis, Arizona State University.

Hung Y, Chen I (1998). A simulation study of dispatch rules for reducing flow times in semiconductor wafer fabrication. *Production Planning & Control* 9: 714-722.

Kim T, Kim H, Lee T, Morrison J, Kim, E (2021). On scheduling a photolithograhy toolset based on a deep reinforcement learning approach with action filter. *2021 Winter Simulation Conference (WSC)*. USA.

Klemmt A, Lange J, Weigert G, Lehmann F, Seyfert J (2010). A multistage mathematical programming based scheduling approach for the photolithography area in semiconductor manufacturing. *Proceedings of the 2010 Winter Simulation Conference*. USA.

Kour G, Saabne R (2014). Real-time segmentation of on-line handwritten Arabic script. *2014 14th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. Greece.

Kour G, Saabne R (2014). Fast classification of handwritten on-line Arabic characters. *2014 6th International Conference of Soft Computing and Pattern Recognition (SoCPaR)*. Tunisia.

Leachman R, Glassey R, Solorzano J (1988). A queue management policy for the release of factory work orders. Research Report. Engineering Systems Research Center, University of California, Berkeley, USA.

Li Y, Jiang Z, Jia W (2014). An integrated release and dispatch policy for semiconductor wafer fabrication. *International Journal of Production Research* 52: 2275-2292.

Mishra S, Prakash, Tiwari M, Lashkari R (2006). A fuzzy goal-programming model of machine-tool selection and operation allocation problem in FMS: A quick converging simulated annealing-based approach. *International Journal of Production Research* 44: 43-76.

Morra J (2017). At ASML, orders pile up for extreme ultraviolet lithography. *Electronic Design*. https://www.electronicdesign.com/technologies/embedded-revolution/article/21805342/at-asml-orders-pile-up-for-extreme-ultraviolet-lithography.

Park S, Fowler J, Carlyle M, Hickie M (1999). Assessment of potential gains in productivity due to proactive reticle management using discrete event simulation. *Proceedings of the 31st Conference on Winter Simulation: Simulation - A Bridge to the Future - Volume 1*. Phoenix, Arizona, USA.

Park I, Huh J, Kim J, Park J (2020). A reinforcement learning approach to robust scheduling of semiconductor manufacturing facilities. *IEEE Transactions on Automation Science and Engineering* 17: 1420-1431.

Peters M, Puharic B (2003). Extending reticle life through better cleaning budgets. *Semiconductor Manufacturing*:89613095.

Raghunathan V (2019). Digital Twins vs simulation: Three key differences. *Entrepreneur*. https://www.entrepreneur.com/article/333645.

Rai R, Kameshwaran S, Tiwari M (2002). Machine-tool selection and operation allocation in FMS: Solving a fuzzy

goal-programming model using a genetic algorithm. *International Journal of Production Research* 40: 641-665.

Randhawa S, Kuo C (1997). Evaluating scheduling heuristics for non-identical parallel processors. *International Journal of Production Research* 35: 969-981.

Randhawa S, Zeng Y (1996). Job shop scheduling: An experimental investigation of the performance of alternative scheduling rules. *Production Planning* & Control 7: 47-56.

Schoolov K (2022). ASML is the only company making the $200 million machines needed to print every advanced microchip. Here's an inside look. *CNBC*. https://www.cnbc.com/2022/03/23/inside-asml-the-company-advanced-chipmakers-use-for-euv-lithography.html.

Simens (2022). PlantSight. https://new.siemens.com/global/en/products/automation/ind-ustry-software/plantsight.html.

Sterling T (2022). Intel orders ASML system for well over $340 mln in quest for chipmaking edge. *Reuters.* https://www.reuters.com/technology/intel-orders-asml-machine-still-drawing-board-chipmakers-look-an-edge.

Sun L, Chen X, Tomizuka M (2014). Selective iterative learning control to deal with iteration-dependent disturbance. *Proceedings Of ISCIE/ASME International Symposium On Flexible Automation*.

Systèmes D (2021). Factory of the future. *Dassault Systemes.* https://www.3ds.com/factory-of-the-future.

Tang C, Denardo E (1998). Models arising from a flexible manufacturing machine, part II: Minimization of the number of switching instants. *Operations Research* 36: 778-784.

Turkcan A, Akturk M, Storer R (2003). Non-identical parallel CNC machine scheduling. *International Journal Of Production Research* 41: 2143-2168.

Turkcan A, Akturk M, Storer R (2007). Due date and cost-based FMS loading, scheduling and tool management. *International Journal of Production Research* 45: 1183-1213.

TWI Global (2022). Simulation vs digital twin (what is the difference between them?). https://www.twi-global.com/technical-knowledge/faqs/simulation-vs-digital-twin.aspx.

Vitelli P (2021). The reticle allocation problem and how to approach it [tech paper review]. *Flexciton*.https://www.flexciton.com/single-post/the-reticle-allocation-problem-technical-paper-review.

Waschneck B, Reichstaller A, Belzner L, Altenmüller T, Bauernhansl T, Knapp A, Kyek A (2018). Deep reinforcement learning for semiconductor production scheduling. *2018 29th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*.

Wikichip.com (2023). Mask / Reticle. *Wikichip*. https://en.wikichip.org/wiki/mask.

Zeballos L (2010). A constraint programming approach to tool allocation and production scheduling in flexible manufacturing systems. *Robotics and Computer-Integrated Manufacturing* 26: 725-743.

Zhang W, Dietterich T (1995). A reinforcement learning approach to job-shop scheduling. *IJCAI* 95: 1114-1120.

Zhang Z, Zheng L, Weng M (2007). Dynamic parallel machine scheduling with mean weighted tardiness objective by Q-Learning. *The International Journal of Advanced Manufacturing Technology* 34: 968-980.

Zhang Z, Zheng L, Li N, Wang W, Zhong S, Hu K (2012). Minimizing mean weighted tardiness in unrelated parallel machine scheduling with reinforcement learning. *Computers* & *Operations Research* 39: 1315-1324.

Chandrasekhar Komaralingam Sivasubramanian is currently a Corporate Quality Systems Professional at Microchip Technology, Chandler. He received a bachelor's degree in mechanical engineering from SRM Institute of Science and Technology, and a master's degree in industrial engineering from Arizona State University in Tempe, Arizona. He is an ASQ Certified Six Sigma Black Belt with 5+ years of manufacturing industry experience.

Robert Dodge is a graduate student currently enrolled in the industrial engineering program at Arizona State University, where he works on topics related to digital twins for smart manufacturing. His primary field of study is in optimization, simulation modeling, and statistical analysis. He began pursing his Ph.D. in 2022 when he was awarded ASU's Dean's Fellowship. Prior to that, Robert graduated with a Bachelor of science in engineering in industrial engineering in 2021, also from Arizona State University.

Aditya Ramani is a mechanical engineering PhD student at Arizona State University specializing in digital twin simulation of critical semiconductor manufacturing processes in collaboration with Intel (where he is currently interning). A prior recipient of the Engineering Graduate Fellowship at ASU, he has 3+ years of industrial experience in the area of software validation and verification testing

in the Aerospace, Car, and Automated Test Systems industries. He has 2 publications focusing in the area of simulation modeling as well as a Bachelors and Masters degrees in Electrical and Aerospace Engineering respectively. He plans to leverage his extensive software experience in performing further research in the area of digital twins and simulations.

David Bayba is a Principal Engineer in Intel's Global Supply Chain Organization. David builds statistical/machine learned, mathematical, and simulation models for supply chain needs at Intel including in the areas of material quality, inventory planning, forecasting, and sourcing intelligence. Prior to that, David supported packaging assembly and test manufacturing at Intel by modeling and establishing equipment and factory performance forecasts. David has been at Intel for 27 years and holds bachelor's and master's degrees in engineering from the University of Arizona and Arizona State University, respectively.

Mani Janakiram is a data and analytics professional at Intel Corporation. His 25+ years of experience includes Semiconductor/Hitech, Automotive and Aerospace industries. He has 2 patents, published 50+ papers and a book on AI. He is an adjunct professor at ASU. He is a PhD in industrial engineering and an MBA. He is a Six Sigma Master Black Belt, is one of the Top 50 Analytics Executive as per CIO.com and an ASCM Fellow.

Eric Butcher is a software research engineer and scientist at Intel Corporation. He is a results-oriented professional with 25+ years of demonstrated experience and innovation in lithography, micro-service architecture, deep litho platform systems development, fault detection and classification (FDC), advanced process control (APC), project management, and integrated production line management. He is passionate about technical writing and mentoring software developers. He serves as a change-agent, driving large scale projects innovating quality and productivity solutions including complex algorithms, leading edge APC, conditional availability, and FDC. He has published multiple papers, and has a Master Degree of science in electrical and computer engineering from Georgia Institute of Technology.

Joseph Gonzales is a semiconductor wafer fabrication professional at Intel Corporation with 15+ years of experience. Having achieved outstanding success in optimization and the development of advanced manufacturing techniques throughout his career, he's most proud of his recent accomplishments in improving lithography utilization improvements. He holds a MS in industrial engineering and a black belt certification in Lean Six Sigma.

Giulia Pedrielli (https://www.gpedriel.com/) is currently associate professor for the School of Computing and Augmented Intelligence (SCAI) at Arizona State University. She graduated from the Department of Mechanical Engineering of Politecnico di Milano. Giulia develops her research in design and analysis of random algorithms for global optimization, with focus on improving finite time performance and scalability of these approaches. Her work is motivated by design and control of next generation manufacturing systems in bio-pharma and aerospace applications, as well as problems in the design and evaluation of complex molecular structures in life-science. Applications of her work are in individualized cancer care, bio-manufacturing, design and control of self-assembled RNA structures, verification of cyberphysical systems. Her research is funded by the NSF, DHS, DARPA, Intel, Lockheed Martin.