Cycle Time Estimation in a Semiconductor Wafer Fab: A concatenated Machine Learning Approach

Zur Erlangung des akademischen Grades eines
DOKTORS DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)

von der KIT-Fakultät für Maschinenbau des
Karlsruher Instituts für Technologie (KIT)
angenommene

DISSERTATION

von

M.Sc. Kai Schelthoff

Tag der mündlichen Prüfung:	18.07.2023
Hauptreferent:	Prof. Dr.-Ing. Kai Furmans
Korreferent:	Prof. Dr. Ivo Adan

Kurzfassung

Die fortschreitende Digitalisierung aller Bereiche des Lebens und der Industrie lässt die Nachfrage nach Mikrochips steigen. Immer mehr Branchen – unter anderem auch die Automobilindustrie – stellen fest, dass die Lieferketten heutzutage von den Halbleiterherstellern abhängig sind, was kürzlich zur Halbleiterkrise geführt hat. Diese Situation erhöht den Bedarf an genauen Vorhersagen von Lieferzeiten von Halbleitern. Da aber deren Produktion extrem schwierig ist, sind solche Schätzungen nicht einfach zu erstellen. Gängige Ansätze sind entweder zu simpel (z.B. Mittelwert- oder rollierende Mittelwertschätzer) oder benötigen zu viel Zeit für detaillierte Szenarioanalysen (z.B. ereignisdiskrete Simulationen). Daher wird in dieser Arbeit eine neue Methodik vorgeschlagen, die genauer als Mittelwert- oder rollierende Mittelwertschätzer, aber schneller als Simulationen sein soll. Diese Methodik nutzt eine Verkettung von Modellen des maschinellen Lernens, die in der Lage sind, Wartezeiten in einer Halbleiterfabrik auf der Grundlage einer Reihe von Merkmalen vorherzusagen. In dieser Arbeit wird diese Methodik entwickelt und analysiert. Sie umfasst eine detaillierte Analyse der für jedes Modell benötigten Merkmale, eine Analyse des genauen Produktionsprozesses, den jedes Produkt durchlaufen muss – was als "Route" bezeichnet wird – und entwickelte Strategien zur Bewältigung von Unsicherheiten, wenn die Merkmalswerte in der Zukunft nicht bekannt sind. Zusätzlich wird die vorgeschlagene Methodik mit realen Betriebsdaten aus einer Wafer-Fabrik der Robert Bosch GmbH evaluiert. Es kann gezeigt werden, dass die Methodik den Mittelwert- und Rollierenden Mittelwertschätzern überlegen ist, insbesondere in Situationen, in denen die Zykluszeit eines Loses signifikant vom Mittelwert abweicht. Zusätzlich kann gezeigt werden, dass die Ausführungszeit der Methode signifikant kürzer ist als die einer detaillierten Simulation.

Abstract

The ongoing digitization of all aspects of life and industry is boosting the demand for microchips. It gets more and more visible that the supply chain of industries – for example the automotive industry – are nowadays dependent on semiconductor manufacturers, which became especially apparent in the ongoing semiconductor crisis. This situation again highlights the need for accurate delivery estimations of semiconductors, but since their production is extremely complex, those estimations remain challenging. Common approaches are either too simplistic (e.g. mean or rolling mean estimators) or need too long for detailed scenario analyses (e.g. discrete-event simulations). Therefore, within this thesis, a new methodology is proposed, which is shown to be more accurate than mean or rolling mean predictors, while being simpler than discrete-event simulations. This methodology makes use of a concatenation of machine learning models, which are able to predict waiting times in a semiconductor fab based feature set. This thesis includes a detailed analysis of the features needed for each model, an estimation of the "route" of a lot through the fab and its impact on the cycle time and developed strategies to cope with uncertainty when feature values are not known in the future. Additionally, the proposed methodology is evaluated with real operational data from a wafer fab of the Robert Bosch GmbH. It is shown that the methodology outperforms mean and rolling mean estimators, especially in situations where a lot cycle time deviates significantly from the mean. In addition, it is shown that the execution time of the method is significantly shorter than that of a detailed simulation.

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit als Doktorand am Institut für Fördertechnik und Logistiksysteme (IFL) des Karlsruher Instituts für Technologie (KIT) in Kooperation mit der Robert Bosch GmbH. Ich möchte mich an dieser Stelle bei allen Personen bedanken, die zum Gelingen dieser Arbeit beigetragen haben.

Ich bedanke mich bei meinem Doktorvater Prof. Dr.-Ing. Kai Furmans, Leiter des IFL, für die Übernahme des Hauptreferats. Seine Forschungen im Bereich der Vorhersage von Zykluszeiten in der Halbleiterindustrie haben mich zu dieser Arbeit inspiriert. Die Möglichkeit zur freien Arbeit und diverse zielführende Diskussionen haben den erfolgreichen Abschluss dieser Dissertation erst ermöglicht.

Prof. Dr. Ivo Adan, Leiter des Lehrstuhls Produktionsnetzwerke an der Technischen Universität Eindhoven, danke ich für die Übernahme des Koreferats.

Ich bedanke mich bei allen aktiven und ehemaligen Kollegen für ihre tatkräftige Unterstützung und die angenehme Arbeitsatmosphäre.

Insbesondere möchte ich mich auf Seiten des IFL bei Dr. Marion Baumann und Christoph Jacobi bedanken, die mir immer mit Rat und Tat zur Seite standen und meinen Einstieg in das Thema mit begleitet haben. Zusätzlich danke ich meinen ehemaligen Masteranden Eva Schlosser und David Plohmann für eine tolle Zusammenarbeit und die Grundsteinlegung vieler Ideen, die in diese Arbeit eingeflossen sind.

Auf Seiten von Bosch haben viele Kollegen zum Gelingen dieser Arbeit beigetragen. Neben der tatkräftigen Unterstützung meiner Kollegen Dr. Georg Laipple, Dr. Robert Kohn, Reiner Moll, Evangelos Angelidis, Dr. Matthias Werner, Dr.

Andrej Gisbrecht und Martin Linder möchte ich insbesondere meinem Betreuer Dr. Michel Janus herzlichst für seine Unterstützung danken. Seine Einführungen in die Welt des maschinellen Lernens und sein Auge für die Details waren richtungsweisend für diese Arbeit.

Ein großer Dank gilt meiner Familie und all meinen Freunden, die an mich geglaubt haben. Insbesondere danke ich meinen Eltern, Dr. Elke Schelthoff und Prof. Dr. Christof Schelthoff, die mich auf meinem gesamten Lebens- und Bildungsweg unterstützt und gefördert haben, und meinem Bruder Tom Schelthoff, der sein großes Wissen immer mit mir geteilt hat und der mir in der Anfertigung meiner Arbeit eine große Unterstützung war.

Mein allergrößter Dank gilt meiner Frau Kerstin van Gaalen für ihre grenzenlose Unterstützung. Sie hat mich immer ermutigt weiterzumachen und sich selbst dabei häufig zurückgenommen.

Karlsruhe, Oktober 2022 Kai Schelthoff

Contents

Acronyms and symbols

Acronyms

ANN	Automatic Material Handling System
ANN	Artificial Neural Network
ARA	Active Routes Approximator
BP	Back propagation
BPN	Backpropagation Network
CART	Classification and Regression Tree
EPT	Effective Process Time
FCFS	First-come-first-served
G	General distribution
GmbH	Gesellschaft mit beschränkter Haftung
HRA	Historic Routes Approximator
IFL	Institute for Material Flow and Logistics Systems
IIOT	Industrial Internet of Things
KPI	Key Performance Indicator
KIT	Karlsruher Institute for Technology

LCFS	Last-come-first-served
MAE	Mean absolute error
ME	Mean error
ML	Machine learning
SC	Semiconductor
RMSE	Root mean squared error
SSE	Sum of squared errors
SSR	Sum of squared residuals

Constants

$\mathbf{e}$	Euler's number

Latin symbols and variables

$\mathbf{A_i}$	inter-arrival time of i-th customer
$\mathbf{c_i}$	work balance
$\mathbf{E[x]}$	expected value
$\mathbf{E_n}$	Error of the n-th value
$\mathbf{e_i}$	occurrence of i-th event of any type
$\mathbf{I_V}$	Variance reduction
$\mathbf{l_max}$	Maximum quantity of lots in any subset
$\mathbf{l_s}$	Subset of lots in subset s
$\mathbf{N}$	number of customers in the system

$\mathbf{N_q}$	number of customers in the queue
$\mathbf{S_i}$	processing time to fulfill i-th customer
$\mathbf{S}$	Feature set before splitting
$\mathbf{S_t}$	Feature set for which the splitting condition applies
$\mathbf{S_f}$	Feature set which does not satisfy the condition
$\mathbf{t_{inc}}$	smallest distinguishable time unit
$\mathbf{T_{on}}$	random variable of the intermediate arrival process
$\mathbf{t_{on}}$	statistical distribution of the inter-arrival time
$\mathbf{T_{off}}$	random variable of the intermediate service process
$\mathbf{t_{off}}$	statistical distribution of the service time
$\mathbf{t_v}$	dwell time
$\mathbf{t_w}$	waiting time
$\mathbf{W_q}$	waiting time of a customer
$\mathbf{w_{ij}^{l-1}}$	weight of incoming connection from layer $l-1$ and node i to node j
$\mathbf{X}$	observed variables
$\bar{\mathbf{X}}$	Mean of a set of values X
$\widetilde{\mathbf{X}}$	Median of a set of values X
$\mathbf{Y}$	target variable
$\hat{\mathbf{Y}}$	estimated values for the target variable

Greek symbols and variables

α	neuron activation function or level of significance
η	learning rate
ϵ	random error of prediction
κ	synapse function
ρ	utilization
γ	inter-arrival-rate
μ	mean processing time/service rate
ν	degree of freedom
$\sigma_{\mathbf{X}}$	Standard deviation of a set of values X
$\sigma_{\bar{\mathbf{X}}}$	Standard error on the mean of a set of values X

Operators and math symbols

$\mathbf{a}$	complex size
$\tilde{\mathbf{a}}$	complex vector

1 Introduction and motivation

> *„In any system in which variable quantities change, it is of interest to examine the effects that some variables exert (or appear to exert) on others.“*
>
> (Draper and Smith 1998, p. 15)

The ongoing digitization of all facets of modern life is increasingly boosting the demand for microchips. Especially in the automotive industry, the demand is constantly on the rise. A recent prognosis from McKinsey (Burkacky et al. (2019)) estimates that the market for electronics and software in cars will evolve from 238 billion $ in 2021 to approximately 469 billion $ in 2030, which is nearly twice as much. It is remarkable that McKinsey even indicates that the software and electronics industry will outgrow the automotive industry by 2030. An example of the immense impact of the semiconductor (SC) industry on the automotive industry is currently given by the ongoing SC crisis. As depicted in Figure 1.1, experts predict that the sales in 2021 will be 5.2 million cars lower, solely because of missing SCs. Additionally, it is apparent that the impact of this crisis will determine the sales of the automotive industry in the upcoming years. As automotive suppliers tend to produce just in time and just in sequence, an accurate estimation of a supply product's completion date is of high need. Due to the ongoing crisis, the supply is currently handled without a lot of stock – which is normally used to buffer delivery uncertainties. There are a variety of metrics that can be used to determine production performance in the SC industry. However, it can be challenging to estimate them depending on the type and nature of the metrics and the manufacturing environment. This is especially relevant in the SC industry, the reasons for which we highlight in the next section.

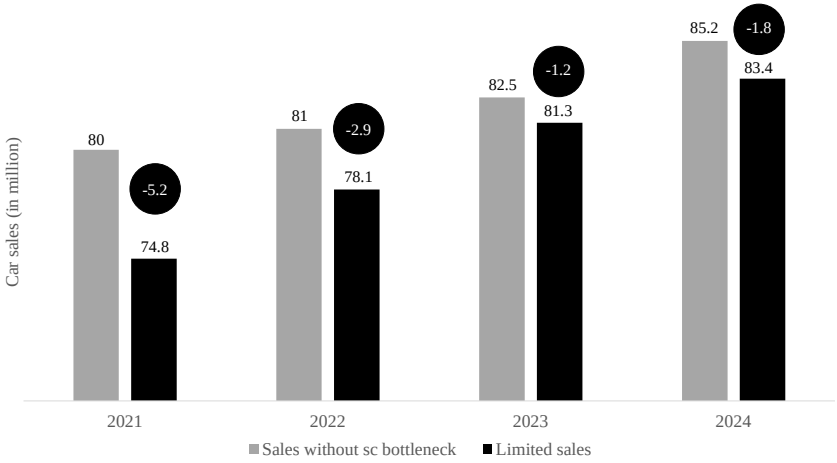


Figure 1.1: Sales prognosis for the automotive industry and the reduction caused by the SC crisis (source: CAR-Auto-Report July 2021)

1.1 Problem description

A lead time estimation can be divided into the production time and the delivery time. While the time to deliver a product depends on the distance to the customer and on the transportation mode – in fact, the delivery time can be rapidly shortened when express delivery is chosen, which is done extensively in the ongoing crisis – the production time cannot be reduced easily. Additionally, the production time cannot be pushed below a certain boundary and can only reach this boundary at the expense of other products production times. Typically, the production of a single SC chip lasts several months, which is a possible explanation for the ongoing crisis, as the SC industry was not able to react quickly to rapid variations in demand (Association (2021)). This is the case due to the following reasons. SC production is characterized by complex and interlinked manufacturing processes. Processing

takes place in loops that can vary in the production processes, depending on the product and the degree of completion of the product. In addition to the cyclical nature of production, bottlenecks due to limited machine capacity emphasize the difficulty of scheduling. Additional challenges in SC manufacturing are the high number of product variations and the high degree of automation (Klemmt 2012, p. 5-15). This increases the importance of the optimization of production in terms of cost structure, flexibility, and productivity (Klemmt 2012, p. 1).

To evaluate a SC factory, key performance indicators (KPIs) are applied, which indicate corporate target measurements like productivity, quality, or sustainability. With the help of these, production can be evaluated against predefined target values. Performance indicators can be differentiated with regard to the evaluation of individual machines and systems or the entire production. An important production-related key figure is the cycle time. The cycle time of a unit incorporates all processing and waiting times (Pfeffer 2012, p. 18-22). Therefore, the cycle time in a SC fab is defined as the time between the entering point qt_{l,n_1} and the leaving point dt_{l,n_N} of a lot l when all necessary operations $n_1, ..., n_N$ are executed:

$$ct_l = dt_{l,n_N} - qt_{l,n_1} \tag{1.1}$$

A closer look at the components of cycle time reveals the waiting time seems to be the most interesting lever in regards to estimating the overall cycle time, since not only planned but also unplanned waiting times occur (Arnold and Furmans 2009, p. 113). Accordingly, when forecasting the cycle time, the forecast of the waiting time is an essential factor, since the waiting time influences the cycle time significantly (Pfeffer 2012, p. 48).

According to Arnold and Furmans (2009), waiting times are often an important factor influencing production planning and can have a wide variety of causes. They can be caused by process-related factors, such as drying or cooling of the product between two process steps, but also by the production process, such as necessary non-value-add steps. Unplanned waiting times are often due to disruptions in the production run due to quality related incidents or changes in the processing sequence (Arnold and Furmans 2009, p. 113). Thus, the waiting time is the

KPI to be used when estimating the cycle time. To be able to estimate it within a production system, several methods are available. Production networks can be modeled as queueing systems. In the discrete-time case, statistical distributions are used to determine the waiting time distributions (Furmans 2004, p. 3).

A second option is the simulation of the production network. In contrast to analytical methods, simulations achieve more accurate results in real-world situations. However, they are significantly more time-consuming and cost-intensive to develop. Furthermore, simulations require high computing times and extensive memory space compared to analytical models (Pfeffer 2012, p. 27f). Despite the high need for accurate delivery estimations, the given estimations are approximative and calculated with simple heuristics (more on that in Section 4.2).

An accurate method for the estimation of the waiting time is needed, which provides results in reasonable computing time. For this purpose, the application of machine learning algorithms is promising. In particular, more advanced estimators like artificial neural networks, random forests or other machine learning methods stand out, which can learn complex relationships and estimate target values (Goodfellow et al. (2018)). Although this field is comparably new, some efforts have already been made. Nonetheless, those efforts either tend to focus on one sub-part of the production process, or use one overall model to estimate cycle times (for example Wang et al. (2021)). While it is valid to estimate only parts of the production process for some applications, it misses validation for whole cycle times. Similarly, it is valid to estimate cycle times with one model, but those approaches are rather inflexible when it comes to the constant changes in a SC fab, which means they are quickly outdated and therefore more applicable for a snapshot of data, and not the operational use. Therefore, we develop a methodology where estimations for sub-parts of the process are concatenated to a complete cycle time estimation. To our knowledge, this is the first approach to estimate cycle time in such a way. Additionally it has to be mentioned that the majority of publications uses artificial data to validate their approaches. In contrast, this thesis provides validation with real operational data from a SC fab.

1.2 Organization of the Thesis

The thesis is organized as follows:

In Chapter 2, we provide an overview about the fundamentals of this work . This part is separated in the main research areas of cycle time estimation: Analytical modeling (Section 2.1), simulation (Section 2.2) and statistical analysis (Section 2.3). Additionally, this part contains a section with definitions of all KPIs used in the course of this thesis (Section 2.4).

In Chapter 3, we summarize current research on the scientific fields introduced in the previous chapter. Since there is also a growing field of approaches where classical methods are combined, a section is included to give an overview on existing literature about hybrid methods (Section 3.4).

In Chapter 4, we explain the SC production process in detail and present industrial standard approaches for cycle time estimation (Section 4.2).

In Chapter 5, we introduce the methodology which is developed in this thesis. Within this approach, we identify three components, which are then elaborated:

1. **Creation of models:** In Chapter 6, the creation and optimization of models for each operation is described. Note that this section was submitted as a scientific paper. Thus, it includes a study, which influencing features are of interest in SC cycle time estimation.

2. **Route estimation:** In Chapter 7, the estimation of the route of a lot is described.

3. **Feature estimation:** In Chapter 8, the estimation of features under uncertainty is presented. Strategies to cope with uncertainties are introduced.

In Chapter 9, we validate the methodology as a whole with real operational data. Within this chapter, the data set is introduced (Section 9.2) and analyzed. Then, a design of experiment is proposed (Section 9.1) and several experiments are executed and analyzed.

In Chapter 10, we summarize and discuss the results, leading to recommendations for the direction of future research in this field.

2 Fundamentals of cycle time estimation

Cycle time estimation is a form of regression analysis, which tries to model the functional dependency of a target value on a set of observables. Applied to the case of cycle time estimation, the dependency of cycle time on different observables of a production environment is modeled. Often this relationship is rather complex and too complicated to describe in simple mathematical functions. If such relationships are found, it is possible to approximate them with mathematical models, which can then be used to estimate future values of this variable (Draper and Smith (1998)). We distinguish between observed variables $(x_1, x_2, ..., x_n)$ and a target variable (Y). Each record in a data set can then be expressed as follows, with vector x as a composition of n features $x_1, x_2, x_3, ..., x_n$ and Y as the target variable, which shall be estimated:

$$(x, Y) = (x_1, x_2, x_3, ..., x_n, Y) \tag{2.1}$$

Regression analysis is then a method to evaluate relationships between features and the target variable (Chung and Huang (2002)). For that, an amount of records (N) is collected. How much records are needed to sufficiently find correlations depends on the complexity of the underlying function and cannot be stated easily, but having more data is leading to better results. According to Chung and Huang (2002), the efforts for cycle time estimations can be distinguished into four categories: Analytical modeling, simulation, statistical analysis, and hybrid methods. The fundamentals of the first three approaches are explained in the following, while hybrid methods make use of the fundamentals of the individual methods and combine them in different ways.

2.1 Analytical modeling

The majority of analytical models in SC manufacturing domain are based on queueing models. Each resource in a production environment is modeled as a node of a network, which in itself reflects the production and its processes. A node of the network is called an operator station and forms an operator system with the upstream waiting room (Furmans (2004)). This is illustrated in Figure 2.1.

In queueing theory, the elements to be processed are called customers, which arrive at service stations based on a random process. In the context of SC manufacturing, customers are therefore lots. Characteristics of a service system can be described using the Kendall notation. This notation follows the scheme *A|B|m - XXXX*, where *A* defines the arrival process and *B* the service process. The number of parallel serving stations is defined by *m*. Customers who are in the waiting room are retrieved according to a serving discipline described by *XXXX*. Well-known serving disciplines are, for example, FCFS (first-come-first-served) or LCFS (last-come-first-served). The way how arrival (λ) and service processes (μ) are modeled, as well as the time steps t between arrivals or service completions considered in the model distinguishes two research areas of queueing theory: Continuous-time and discrete-time, which are explained in the following.

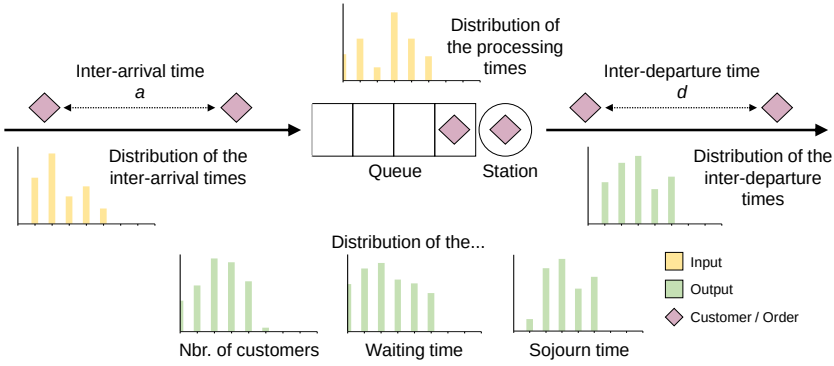


Figure 2.1: Elements of a queueing theory model; following (Furmans 2004, p. 3)

2.1.1 Continuous-time queueing theory

In the continuous-time queueing theory, t can take any value from $T = [0; \infty)$. The probability distributions of the intermediate arrival and the service time are typically represented as continuous functions by an exponential distribution. Their density function is:

$$f(t) = \lambda e^{-\lambda t} \tag{2.2}$$

The corresponding distribution function is:

$$F(x) = \int_0^x \lambda e^{-\lambda t} dt = 1 - e^{-\lambda x} \ \forall x \geq 0 \tag{2.3}$$

For a queueing system with one queueing station, if inter-arrival and queueing times are both exponentially distributed, the state of the queueing system can be fully expressed with the number of customers in the system. The expected number of customers in the system can solely be expressed with the utilization ρ, which is defined as the fraction of arrival rate λ and the service rate μ:

$$\rho = \frac{\lambda}{\mu} \tag{2.4}$$

Subsequently, the expected number of customers in the system is obtained:

$$N = \frac{\rho}{1 - \rho} \tag{2.5}$$

For the expected value of the number of customers waiting N_q the following holds true:

$$N_q = \frac{\rho^2}{1 - \rho} \tag{2.6}$$

Little‘s law can then be used to link the expected number of customers with the expected waiting time of a customer W_q:

$$N_q = \lambda \cdot W_q \tag{2.7}$$

The expected waiting time can therefore be expressed as the fraction of expected number of customers in the queue N_q and the arrival rate λ, and consequently the fraction of the number of customers in the system and the mean processing time μ (Waldmann and Helm (2016)) :

$$W_q = \frac{N_q}{\lambda} = \frac{N}{\mu} \tag{2.8}$$

The derivation of the equations above can be seen in detail in Furmans (2000).

The previously described relationships are a simplified way to determine the waiting time. Among other things, the condition has to be served that intermediate arrival and service times are exponentially distributed. Nevertheless, continuous-time queueing models are able to calculate general distributions as well, but only approximately. In general, for the continuous-time queueing theory the condition applies that we approximate the probability distributions by suitable functions (Arnold and Furmans 2009, p. 148). If this assumption cannot be made, discrete-time queueing models are the method of choice, which are introduced in the following.

2.1.2 Discrete-time queueing theory

In most real-life scenarios concrete distributions can only be assumed approximately for the arrival and service process. Therefore, a methodology is needed which is capable of handling all sorts of "discrete" distributions. In the discrete-time theory of operation, events in the system are considered to occur at regular intervals t_{inc}, which is the smallest distinguishable time unit so that intermediate arrival and service time distributions are multiples of this time unit. Intermediate arrival and service processes are defined as renewal processes, as the respective distributions do not necessarily have to be exponentially distributed (Arnold and Furmans 2009, p. 152). According to Schleyer and Furmans (2007) the discrete-time queueing theory allows both a higher accuracy of the results and a higher level of detail compared to continuous-time queueing theory.

The statistical distribution of the inter-arrival time is defined by t_{on}. Thus, the expected arrival rate $\lambda = 1/E[t_{on}]$ can be determined, which is the inverse of the expectation value of the intermediate arrival time. Accordingly, the service time t_{off} of a customer at a station is also characterized by a statistical distribution, the service time distribution. Following the arrival rate λ, the service rate is defined as $\mu = 1/E[t_{off}]$ and denotes the average number of customers that can be served at a service station at full capacity given the distribution of service time (Arnold and Furmans 2009, p. 112f).

To calculate the distributions of waiting time, sojourn time and number of customers from the known distributions of arrival and service time, the model must be able to reach a state of equilibrium. This is possible if the load factor $\rho = \lambda/\mu$ is less than 1. Only then the average number of arriving customers per time unit is smaller than the expected number of served customers per time unit. If $\rho \geq 1$ holds, the queueing system cannot reach an equilibrium state and the queue would grow faster than customers can be served (Arnold and Furmans 2009, p. 113f). Compared to continuous-time queueing models, discrete-time models are capable of dealing with all general distributions G. This results in the notation *G|G|1* for discrete-time control systems with one control station (Furmans 2000, p. 9).

If a station is still occupied by another customer when a customer arrives, the arriving customer must wait in the waiting room. This waiting time t_w also follows a probability distribution. Accordingly, the dwell time t_v is composed of the waiting time t_w and the service time t_{off} (Furmans 2000, p. 8f).

On a discrete time axis, a series of events with time intervals between event $n-1$ and n is given. This represents a renewal process if the lengths of all time intervals are independent of each other and follow the same distribution (Schleyer and Furmans 2007, p. 9f).

With these assumptions, probability distributions of the intermediate arrival and service processes can be represented using the random variables T_{on} and T_{off}:

$$\begin{aligned} P(T_{on} = j \cdot t_{inc}) &= t_{on|j} \qquad \forall j = 1, ..., j_{t_{on},max} \\ P(T_{off} = j \cdot t_{inc}) &= t_{off|j} \qquad \forall j = 1, ..., j_{t_{off},max} \end{aligned} \tag{2.9}$$

By assuming that inter-arrival and service time distributions are independent of each other, the new random variable X is created with its corresponding realization, as follows:

$$X^{(n)} = T_{off}^{(n)} - T_{on}^{(n)} \tag{2.10}$$

t_{on} and t_{off} can be transformed into probability vectors to calculate the system's work balance. The work balance denotes the change in the system's labor supply due to the arrival of a new customer and is calculated by convolving $t_{on|j}$ and $t_{off|j}$:

$$c_i = \sum_{j=0}^{\infty} t_{on|j} \cdot t_{off|i-j} \tag{2.11}$$

Once the work balance c_i has been calculated, the probability distribution of the waiting times of the customers is calculated. Here, a distinction is made between waiting times that are greater than zero and waiting times that are equal to zero. Waiting times greater than zero are denoted by:

$$w_i = \sum_{j=-j_{t_{on},max}}^{j_{t_{off},max}} w_j \cdot c_{i-j} \qquad \forall i = 1, 2, ... \tag{2.12}$$

Waiting times equal to zero can occur in two situations. Either the departure of the customer coincides with the arrival of the next customer at the service station, or the respective queueing systems are empty, which can happen when the supply is short and therefore a fallow time is formed (Arnold and Furmans 2009, S. 150-154):

$$w_0 = \sum_{k=-j_{t_{on},max}}^{j_{t_{off},max}} w_k \cdot c_{-k} \qquad + \sum_{k=0}^{\infty} w_k \cdot \sum_{l=-j_{t_{on},max}}^{-(k+1)} c_l \tag{2.13}$$

It must be noted that a method is available to significantly speed up this calculation, using the numerical solution of Grassmann and Jain (Grassmann and Jain (1989)).

2.2 Simulation

As an alternative to analytical models simulations are used, which – in oppose to an analytical model – try to replicate the entities of a system, instead of solely their distributions. They are one of the most popular methods in operations management (Lane et al. (1993) and Gupta (1997)). Within simulations, a collection of entities is modeled as a system, which then takes a certain state. A state is defined as a collection of variables to describe the system at a particular time. Two types of systems are distinguished, discrete and continuous. (Law and Kelton (2000)) In a continuous system, the state variables change continuously over time, while in discrete systems those variables change only on certain points in time. According to Law and Kelton (2000), simulation models can be classified by three dimensions:

- *Static vs. Dynamic Simulation Models.* While static simulation models are aiming to represent a system at a specific time point, dynamic simulation models are aiming to represent the development of a system over time.
- *Deterministic vs. Stochastic Simulation Models.* A deterministic simulation model does not cover any probabilistic components. Most systems however cover at least some parameters with probabilistic influence and are therefore captured in stochastic simulation models.
- *Continuous vs. Discrete Simulation Models.* Dynamic systems may be continuous or discrete. Continuous dynamic systems (e.g. physical systems with moving material objects) are defined by state variables whose values continuously change. In opposition, state variable values of discrete dynamic systems (e.g. manufacturing simulations) are changed at discrete time steps only (Birta and Arbez 2013, pp.249).

Since most models in the field of cycle time estimation in SC manufacturing are discrete, dynamic and stochastic, only those fundamentals are covered in the following. All models within this three-dimensional area are called *discrete-event*

simulation models.

Discrete-event Simulation

Within discrete-event simulations, each event (e.g. a start of production of a lot) triggers a change of the system state at a particular time point. To do that, one key element of a discrete-event simulation model is a proper time-advance mechanism. The corresponding variable in a simulation model which keeps track of the simulated time is called simulation clock (Law and Kelton 2000, p. 7). In general, two approaches for advancing the simulation clock have been suggested: Next-event time advance and fixed-increment time advance. While the fixed-increment time advance is based on a fixed interval in which all variables are updated, the next-event time advance jumps forward to certain events and updates the variables at those time points. Between those points in time, the system is assumed to remain stable, which allows the system to directly jump to the event times. Hence, it can estimate future behavior on a shorter timescale (Robinson (2014)). Since nearly all common simulation softwares and models are using the next-event advance mechanism, we will focus on this technique in the following.

Within this approach, the simulation clock is initialized with zero and the time of future events is calculated as the timely distance from it. Then, the simulation clock is advanced to the first future event, where the system state is then updated. The process is visualized in Figure 2.2, with t_i as the time of arrival of the i-th customer, $A_i = t_i - t_{i-1}$ as the inter-arrival time, S_i as the processing time for the fulfillment of the i-th customer, c_i as the completion time of the ith customer and e_i as the occurrence of the i-th event of any type. Hence, the events where the times have to be updated can be reduced to any arrival and completion times. In general, each of those quantities is probabilistic and can be therefore modeled as a random variable. Mostly, the probability distributions of the inter-arrival times and the processing times are assumed to be known.

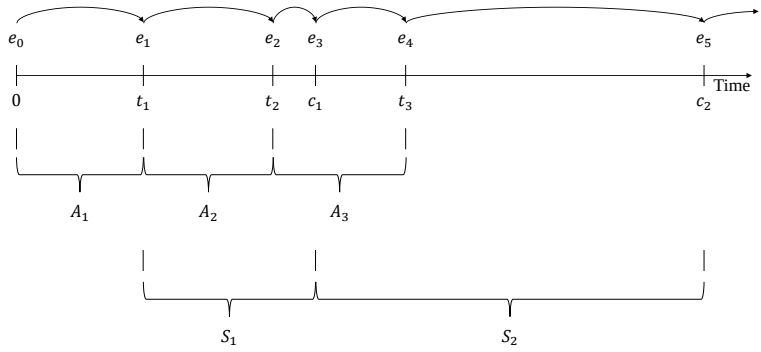


Figure 2.2: The next-event time-advance approach for a single-server queueing system ((Law and Kelton 2000, p. 9))

Discrete-event simulations can model concurring events like simultaneous arrivals of lots at work stations. To do that, the system behavior is calculated for every time step sequentially. Thus, it is capable of covering any desired complexity, but it can be very calculation-intensive (Shanthikumar et al. (2007)). Therefore, when combined with the event uncertainty, it is nearly impossible to simulate all possible events in a fab, while being faster than real-time.

2.3 Statistical Analysis

Another method to estimate cycle time is statistical analysis, which makes use of known statistical distributions in a system to estimate unknown ones. In statistical analysis mathematical models are built that estimate the relationship between one or more variables and one or more target variables. Those relationships are usually used to either explain past behavior, or – more often – to estimate future behavior. There are countless approaches to determine such relationships, but in the context

of this thesis, we focus on Machine Learning (ML) approaches. In the following, we introduce their fundamentals as well as popular ML approaches.

2.3.1 Machine Learning

ML is a form of applied statistics. It differs from classical statistics in its focus on using existing computational capabilities „to statistically estimate complicated functions“ (Goodfellow et al. 2018, S. 107). Thus, ML refers to the ability of computers or artificial intelligence to recognize patterns in data and generate knowledge from them through generalization (Goodfellow et al. 2018, p. 3). The process of generating knowledge from data is called "training", in which typically only a fraction of the available data is used, so that the performance of the model with unseen data can be evaluated. This performance evaluation is called "test". Typically, a third fraction of data is held back for the "validation" of the model.

ML can be categorized as "supervised, unsupervised, reinforcement, and evolutionary" learning. In supervised learning, the algorithm receives training data and target values. It is supposed to learn patterns and relationships from the training data and use this knowledge to estimate the corresponding target data. Unsupervised learning algorithms, on the other hand, receive only training data and are expected to recognize and categorize relationships between these inputs. Reinforcement learning is a hybrid between supervised and unsupervised learning, and evolutionary ML algorithms are based on continuous adaptation. The most common application is in supervised learning algorithms. These can be used either for regression or for classification of the data. In classification, the target values correspond to classes into which the training data should be divided. This involves deciding which values of the inputs belong in which class. A classification problem is always discrete. In a regression, the algorithm receives training data containing various features and target values. Based on relationships in the training data that the algorithm learns, the target value is to be estimated (Marsland 2015, S. 6ff). Since the target of this study is to develop a regression

method for cycle time, we focus on regression applications for the following ML techniques – while they are in principle also valid to use for classification.

The goal of ML methods is to generalize, which means that the model develops the ability to produce sensible outputs from unknown input. There are two phenomenons known, which can result in poor generalization capabilities of ML models: Underfitting and overfitting. When a model is underfitting it means that it has not learned to anticipate patterns in the data during training. Thus, a model which is underfitted does not react sensitively to the input data, which is often times the result of too short training. The algorithm was not able to generalize enough to produce estimations with high accuracy. In opposition, a model which reacts too sensitively to the input data it has already seen is called overfitted. In this case, the algorithm learns not only patterns but also noise and outliers of the training data and is no longer able to generalize. This is often a consequence of too long training (Marsland (2015)). The best known indicator of the generalization capabilities of a model is the comparison of the estimation performance on the training data compared to the previously unseen test data. When the performance on the training data is significantly higher than the performance on the test data, the model is most probably overfitted. If the performance on the train data and the test data is bad, the model could be underfitted.

In the following, we present four popular ML techniques: Linear Regression, ANNs, Decision Trees, and Random Forests. For each of those techniques, we also discuss the risk of under- and overfitting.

2.3.1.1 Linear Regression

Linear Regression is a popular method, because its results are easy to interpret, since it is possible to show the parameter search by deriving the cost function to each parameter. To apply it, the assumption has to be made that the assumed functional relationship between the observed variable X and the target variable

Y is linear. Hence, we assume that we can describe the target variable dependent on the observed variable:

$$Y = \beta_0 + \beta_1 X \tag{2.14}$$

It must be noted that this is valid for the one-dimensional case. For more than one variables, all variables are vectors. Thus, Y can be represented by a linear transformation of X plus an amount ϵ, which indicates the random error of such an estimation (Draper and Smith (1998)). This means, that β_0, β_1 and ϵ are unknown. ϵ is not steady and changes for each observation. To determine β_0 and β_1, we introduce estimations for both parameters (b_0 for β_0 and b_1 for β_1). With those estimated parameters, we can generate estimations $\hat{Y}$ for Y:

$$\hat{Y} = b_0 + b_1 X \tag{2.15}$$

Between every estimation and true value, there is always a non-explainable error ϵ different from zero. To estimate the parameter values, the least squares estimation is commonly used. Other methods possible are minimizing the lack of fit with other norms (e.g. least absolute deviation), or minimizing a penalized version of the cost function (e.g. L^2-norm penalty). The least squares method finds optimal parameter values by minimizing the sum of squared residuals (SSR):

$$SSR(\beta_0, \beta_1) = \sum_{n=1}^{N} (Y_n - \hat{Y}_n)^2 = \sum_{n=1}^{N} (Y_n - b_0 - b_1 X_n)^2 \tag{2.16}$$

It is now the task to vary b_0 b_1 so that SSR gets as small as possible. By differentiating Equation (2.16) we can determine optimal b_0 and b_1 :

$$\frac{\partial SSR}{\partial b_0} = -2 \sum_{n=1}^{N} (Y_n - b_0 - b_1 X_n) \tag{2.17}$$

$$\frac{\partial SSR}{\partial b_1} = -2 \sum_{n=1}^{N} X_n (Y_n - b_0 - b_1 X_n) \tag{2.18}$$

Hence, b_0 and b_1 are solutions of the following equations:

$$\sum_{n=1}^{N}(Y_n - b_0 - b_1 X_n) = 0 \tag{2.19}$$

$$\sum_{n=1}^{N} X_n(Y_n - b_0 - b_1 X_n) = 0 \tag{2.20}$$

This basic technique of finding the optimal set of parameter values is also used by other ML methods, or more advanced search techniques through the parameter space.

Since Linear Regression is only capable of finding linear patterns in the training data, the method is of high risk of underfitting, while the risk of overfitting is comparably low.

2.3.1.2 Artificial Neural Networks

Another popular approach in ML is ANN, which represents any collection of inter-connected neurons, with each neuron producing a certain response at a given set of input signals. By applying an external signal to some (input) neurons the network is put into a defined state that can be measured from the response of one or several (output) neurons. Hence, an ANN can be seen as a mapping from a space of the input variables $x_1, x_2, ..., x_N$ to a one- or multi-dimensional space of output variables $y_1, y_2, ..., y_M$. In the given case, just one output variable (the sojourn- or cycle time) is needed. Its behavior is determined by the layout of the network, which consists of the number of neurons and their relationship to each other, the weights of the inter-neuron connections, and by the response of the neurons to the input, which can be described by the neuron response function ρ.

In principle, a neural network with n neurons can have $(n-1)^2$ connections (not n^2 because a neuron can not be connected to itself). Nevertheless, the complexity can be reduced by organizing the neurons in layers, allowing only connections

from a layer to its direct successor. This structure is called "Multilayer Perceptron" (MLP) and is visualized in Figure 2.3. Within an MLP, the first layer is called the "input layer", and the last one the "output layer", while all others are called "hidden layers". Within a regression problem – similar as for classification problems – with N input variables, the input layer consists of N neurons that hold the input values $x_1, x_2, ..., x_N$ and one neuron in the output layer representing the output variable, the estimated target variable of the ANN y_{ANN}. Note that is it theoretically possible to have more than one output variable, for example for a multi-target regression, but since this is not the case in this thesis, its theory and implications are left out of this summary.

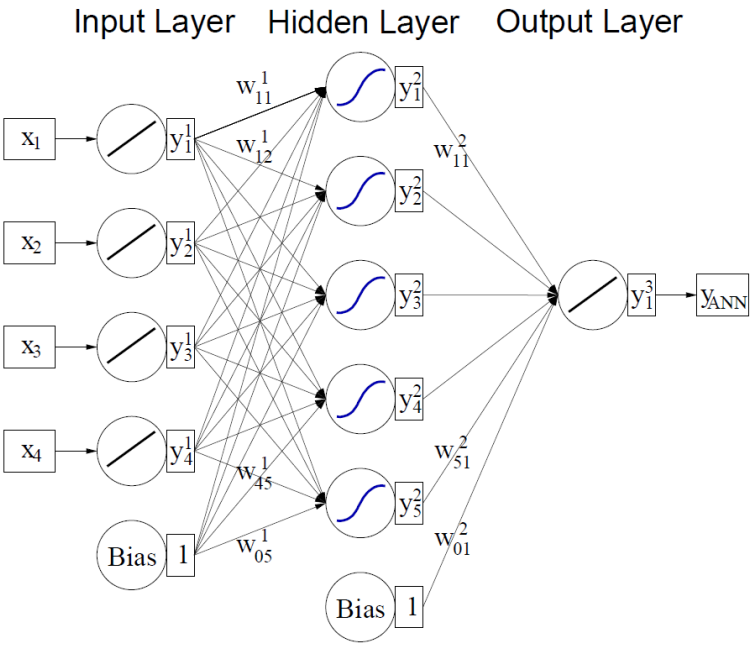


Figure 2.3: Multilayer perceptron with one hidden layer, with linear activation function in the input and output layers, and sigmoid activation function in the hidden layer. (Hocker et al. 2007, p. 110).

The neuron response function p is used to map the neuron input $i_1, i_2, ...i_N$ onto the neuron output, as depicted in Figure 2.4. It can be separated into a $\mathbb{R}^n \mapsto \mathbb{R}$ synapse function κ, and a $\mathbb{R} \mapsto \mathbb{R}$ neuron activation function α, so that $p = \alpha \circ \kappa$. These functions can have the following forms:

$$\kappa : (y_1^{(l)}, ..., y_N^{(l)} | w_{0j}^{(l)}, ..., w_{Nj}^{(l)}) \rightarrow \begin{cases} w_{0j}^{(l)} + \sum_{n=1}^{N} y_n^{(l)} w_{nj}^{(l)} & \text{Sum,} \\ w_{0j}^{(l)} + \sum_{n=1}^{N} (y_n^{(l)} w_{nj}^{(l)})^2 & \text{Sum of squares,} \\ w_{0j}^{(l)} + \sum_{n=1}^{N} |y_n^{(l)} w_{nj}^{(l)}| & \text{Sum of absolutes,} \end{cases} \tag{2.21}$$

$$\alpha : x \rightarrow \begin{cases} x & \text{Linear,} \\ \frac{1}{1+e^{-kx}} & \text{Sigmoid,} \\ \frac{e^x - e^{-x}}{e^x + e^{-x}} & \text{Tanh,} \\ e^{-x^2/2} & \text{Radial.} \end{cases} \tag{2.22}$$

Note that – as visible in the example of Figure 2.3 – it is common to use different

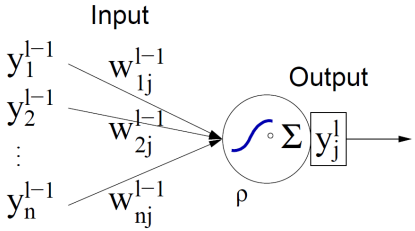


Figure 2.4: Single neuron j in layer l with n input connections. Each incoming connections carries a weight $w_{ij}^{(l-1)}$ (Hocker et al. 2007, p. 110)

activation functions for the visible and the hidden layers, to enable the model to cover multiple kinds of correlations.

When building a network, the Stone-Weierstrass theorem has to be considered. It states – when applied to neural networks – that for an MLP a single hidden

layer is sufficient to approximate any continuous correlation function to optimal precision, given the fact that a sufficient amount of neurons is used in the hidden layer (De Branges (1959)). Hence, if the computing power and the amount of training data allows, one can increase the number of neurons until the optimal performance is reached. Nevertheless, it is likely that a similar performance can be reached with fewer neurons and more hidden layers, which potentially decreases training time and leads to more robust networks.

Training of an MLP

Learning patterns in data is done by training the algorithm using training data. The training process is explained in the following. During training, the goal is to set the weights of the nodes in an optimal way, so that a cost function is minimized. For regression problems, an error measure like the sum of squared errors is often used. The most common algorithm to adjust the weights is the so-called "back propagation" (BP). It is a supervised-learning technique, where the desired output is known (e.g. for each lot in training data, the cycle time is known). To illustrate how the training process works, a simple MLP with only one hidden layer and a Tanh activation function in the hidden layer is assumed. For this MLP, the output of this network is given by:

$$y_{ANN} = \sum_{j=1}^{n_h} y_j^{(2)} w_{j1}^{(2)} = \sum_{j=1}^{n_h} tanh(\sum_{i=1}^{n_{var}} x_i w_{ij}^{(1)}) \dot{w}_{j1}^{(2)} \tag{2.23}$$

with n_{var} as the number of neurons in the input layer, n_h as the number of neurons in the hidden layer, $w_{ij}^{(1)}$ as the weight between input-layer-neuron i and hidden-layer-neuron j, and $w_{i1}^{(2)}$ as the weight between the hidden-layer-neuron j and the output neuron. Within this equation, the sum was used for the synapse function κ.

Within training, the network is fed with K training events $x_a = (x_1, ..., x_{n_{var}})_a$, $a = 1, ..., K$. For each event a, the output $y_{ANN,a}$ is computed and compared to the desired output y_a. An error function E measures the agreement of the network output with the desired one. There are several possible error functions,

but a typical error function is the sum of squared errors (SSE) and is defined as follows:

$$SSE((x_1, ..., x_K)|w) = \sum_{a=1}^{K} SSE_a(x_a|w) = \sum_{a=1}^{K} \frac{1}{2}(y_{ANN,a} - y_a)^2 \quad (2.24)$$

with w as the set of weights in the network. During training, the goal is to find the set of weights w that minimizes the error function. This can be reached using the "gradient descent" (assuming that the neuron response function is differentiable with regard to the input weights). Starting from a set of weights $w^{(\rho)}$, which could be either chosen randomly or from past or similar models, the weights are constantly updated by moving them in w-space into the direction $-\nabla SSE$, where SSE decreases the most.

$$w^{(\rho+1)} = w^{(\rho)} - \eta \nabla_w SSE \quad (2.25)$$

with the positive number η as the "learning rate", which represents the moving speed in the w-space.

The weights connected with the output layer (in this case the ones from the hidden layer) are updated by:

$$\Delta w_{ij1}^{(2)} = -\eta \sum_{a=1}^{K} \frac{\partial E_a}{\partial w_{j1}^{(2)}} = -\eta \sum_{a=1}^{K} (y_{ANN,a} - y_a) y_{j,a}^{(2)} \quad (2.26)$$

The weights connected with the hidden layers (in this case the ones from the input layer) are updated by:

$$\Delta w_{ij}^{(1)} = -\eta \sum_{a=1}^{K} \frac{\partial E_a}{\partial w_{ij}^{(1)}} = -\eta \sum_{a=1}^{K} (y_{ANN,a} - y_a) y_{j,a}^{(2)} (1 - y_{j,a}^{(2)}) w_{j1}^{(2)} x_{i,a} \quad (2.27)$$

Note that in this simple example $tanh'(x) = tanh(x)(1 - tanh(x))$ is used. This method is called "bulk learning", since the sum of errors of all training events

is used to update the weights. Another possibility would be "online learning", where the weights are updated after each event.

BP is not the only possible method to train a model. Other options, such as the "Broyden-Fletcher-Goldfarb-Shannon" method, will not be covered in this thesis, since it is not used and therefore of minor interest.

When BP is executed until a predefined value of error function E (accuracy) is reached, it leads to a set of weights w^{ρ}_{opt}. As the Weierstrass theorem indicates, it would be theoretically possible to reach any desired accuracy, when enough data and computational power is available. However, since data and computational power are limited resources, this cannot be achieved all the time. Additionally, as mentioned before, running the training long imposes the risk of overfitting to the model, while too short training runs could result in underfitting. Thus, the training of an ANN is a balancing act of those two extremes.

2.3.1.3 Decision Trees

A decision tree is a classical ML algorithm, which applies a defined number of splitting criteria to data sets to estimate a target variable, as shown on the exemplary decision tree depicted in Figure 2.5. Decision trees are commonly used for two tasks in data mining: First, to estimate whether an element can be assigned to a certain class (classification), which means that the output possibilities are a discrete number of classes (e.g. whether an image shows a dog or a cat). Second, to estimate a real number (regression), which means that the output can be any real number (e.g. the estimation

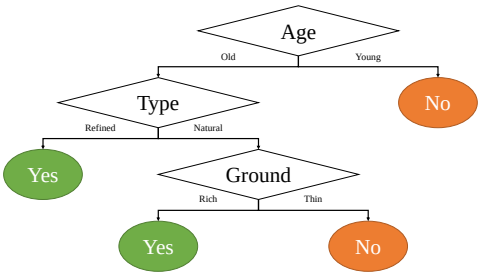


Figure 2.5: A binary decision tree to estimate whether an apple tree will bear fruit

of a stock price in the future). Solving techniques for both problems with random forests can be summarized with *Classification and Regression Tree* (CART) and were first introduced by Breiman et al. (1984).

Training of a decision tree is called "growing". A good introduction is provided by Janus (2013), and can be done using the following algorithm:

1. Iteration over the list of input variables. A cut value is determined that gives the best separation between two classes of outputs (e.g. if a cycle time is higher or lower than a certain value).

2. The input variable that gives the best separation is chosen and the sample of events is separated into two daughter nodes, which are called "leaves" according to the cut on this variable.

3. Steps 1 and 2 are repeated on each of the daughter leaves until a stopping criterion is reached. This stopping criterion is usually a minimum number of events of each type in one leave.

4. If the stopping criterion is reached for a leaf, the output of the leaf is defined as the median of all events being in this class (e.g. a certain cycle time).

Within this process, one input variable can be used at multiple points, since used variables are not erased from the input set. Hence, the sequence of cuts from the root to each leave defines a hyper cube in the N-dimensional space of the input variables $x_1, ..., x_N$.

This is based on the principle of variance reduction. It tries to reduce the variance of the subsets by splitting the data set at a node N. The variance reduction I_V of the target variable y can be calculated as follows:

$$\begin{aligned} I_V(N) = \frac{1}{|S|^2} \sum_{i \in S} \sum_{j \in S} \frac{1}{2}(y_i - y_j)^2 \\ -\left(\frac{1}{|S_t|^2} \sum_{i \in S_t} \sum_{j \in S_t} \frac{1}{2}(y_i - y_j)^2 + \frac{1}{|S_f|^2} \sum_{i \in S_f} \sum_{j \in S_f} \frac{1}{2}(y_i - y_j)^2\right) \end{aligned} \tag{2.28}$$

S corresponds to the set before splitting, S_t corresponds to the set for which the splitting condition applies, and S_f to the set which does not satisfy the condition. (Han et al. 2011, p. 332ff) While trees used for classification and trees used for regression have some similarities, they differ, for example in their procedure where to split (Breiman et al. (1984)).

Decision trees are capable of covering as many hyper cubes as possible – given a sufficient depth of the tree. This means that decision trees can estimate any volume in the variable space, and therefore any type of correlation. On the other side, decision trees tend to be unstable when statistical fluctuations in the training samples occur. Additionally, when the tree is built sufficiently deep, the risk of overfitting is significant. To circumvent these problems, several methods have been developed, which make use of so-called "ensemble learning", meaning the creation of an ensemble of trees. Those trees are combined to make an estimation – or classification. Three ensemble techniques can be mentioned. Firstly, a technique called "boosting" (Freund et al. (1996)), using the "AdaBoost" algorithm (Freund and Schapire (1997)), where a succession of decision trees with the same inputs and samples is built. For each successive tree, the falsely estimated events by the previous tree are weighted with a so-called "boost weight", to ensure that the estimation of those events will be handled with priority in the next tree. The weights are then renormalised so that the sum of weights remains constant. Ultimately, any event is then estimated by the median of all tree decisions. Secondly, a technique called "bootstrap aggregated" decision trees could be applied, where multiple decision trees are built by resampling the training data (Breiman (1996)). In oppose to boosted trees, bootstrap aggregated trees use random samples of the data to build successive trees, instead of the whole data set every time. Thirdly, "rotation forests", where a principal component analysis is applied on a random subset of the input features, before a tree is trained (Rodriguez et al. (2006).)

2.3.1.4 Random Forests

Random Forests were first developed by Ho (1995), and are a specific type of bootstrap aggregated decision trees. They were further developed by Breiman (2001), who also registered "Random Forests" as a trademark in 2006. Random Forests – as depicted in Figure 2.6 – combine a large number of decision trees to reduce the variance in their solution quality (Couronné et al. (2018)) and overcome the poor generalization capability of single random forests (Huang et al. (2016)). For a random forest, each CART is built on a random bootstrap data sample from the original data set. This is done B times. Hence, the process can be expressed as follows:

For $b = 1, ..., B$:

1. Sample, with replacement, n training samples, which are called X_b, Y_b.
2. Train a tree f_b, as described in Section 2.3.1.3.

Afterwards, the estimations for the samples which have been unseen in training (x') can be made by averaging the estimations from all trees:

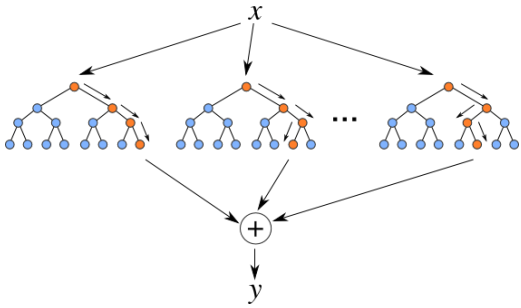


Figure 2.6: Depiction of a regression random forest (Bakshi (2020))

$$\hat{f} = \frac{1}{B} \sum_{b=1}^{B} f_b(x') \tag{2.29}$$

Subsequently, a random forest can be expressed as follows, with $t(x, s_{\Theta_k})$ as the base estimator, which is represented by a CART $k = 1, 2, ..., m$, using the input vector x:

$$\{t(x, s_{\Theta_1}), t(x, s_{\Theta_2}), ..., t(x, s_{\Theta_m})\} \tag{2.30}$$

s_{Θ_k} is a random vector, representing the random extraction of training samples for the kth tree. Hence, all s_{Θ_m} share the same distribution, but are independent from each other. Additionally, as only a fraction of features is considered at each split when building each tree, the growth process also represented by s_{Θ_k} is also random.

While all the above holds true for all bootstrap aggregated trees, random forests make additional usage of a technique called "feature bagging". This technique uses only a random subset of features at each split to avoid that strongly correlated features are used too extensively, which can easily lead to over-fitting (Ho (2002)).

Random forests are a popular technique in businesses, since they generate useful estimations with low configuration efforts. Compared to Decision Trees, they are less prone to overfitting, since feature bagging counteracts that. Nevertheless, they loose one of the biggest advantages of decision trees: their interpretability.

2.4 Definitions of statistical properties

To compare all approaches presented in this thesis, KPIs have to be introduced, which are defined in Section 2.4.1. To prove the statistical significance of the KPI deviations, statistical hypothesis tests are used. Their fundamentals are introduced in Section 2.4.2.

2.4.1 Definitions of KPIs

As mentioned in Chapter 2, we observe a target variable Y_n. When an estimation has been made, a set corresponding estimated values $\hat{Y}_n$ is computed. Each performance metric compares Y_n and $\hat{Y}_n$ for each $n = 1, 2, ..., N$. The most intuitive comparison is to check standard positional parameters of both the target

variable distribution as well as the estimation distribution. First, the mean $\bar{X}$ of a set of values $X_1, X_2, ..., X_N$ is compared and can be calculated as follows:

$$\bar{X} = \frac{1}{N} \sum_{n=1}^{N} X_n \tag{2.31}$$

If the mean of the target variable distribution and the estimation distribution differs, a systematic shift in the distribution is observed. Since the mean is an indicator of the position of the center of the distribution, another positional parameter for the width of the distribution is needed. Subsequently, the unbiased standard deviation of the sample σ_X can be considered:

$$\sigma_X = \sqrt{\frac{1}{N-1} \sum_{n=1}^{N} (X_n - \bar{X})^2} \tag{2.32}$$

Because $X_1, X_2, ..., X_N$ is a sample of values from an unknown distribution, it is important to mention that the actual positional parameters of the distribution are not known. Therefore, the observed mean $\bar{Y}$ and the observed standard deviation σ_X do not necessarily equal the mean of the distribution μ and the standard deviation of the distribution σ. Hence, the mean value of this sample $\bar{X}$ comes along with a *standard error on the mean* $\sigma_{\bar{X}}$:

$$\sigma_{\bar{X}} = \frac{\sigma}{\sqrt{N}} \tag{2.33}$$

Because σ is not known, it is usually estimated by the sample standard deviation σ_X:

$$\sigma_{\bar{X}} \approx \widehat{\sigma}_{\bar{X}} = \frac{\sigma_X}{\sqrt{N}} \tag{2.34}$$

Additionally, the median of both distributions could be compared. The median $\widetilde{X}$ can be calculated as follows, with m as the middle observation of a distribution:

$$\widetilde{X} = \begin{cases} X_{m+1} & if\ N\ is\ odd \\ \frac{1}{2}(X_m + X_{m+1}) & if\ N\ is\ even \end{cases} \tag{2.35}$$

The median is the value of a distribution where 50 % of the values are below this threshold. Hence, it is a positional parameter that is more robust against outliers in the distribution. All the above positional parameters can be calculated for the observed values ($X = Y$ and $X_n = Y_n$) as well as the estimated values ($X = \hat{Y}$ and $X_n = \hat{Y}_n$), which enables a comparison of both.

In addition to the comparison of positional parameters, error parameters are considered. In general, an estimation error for the n-th value E_n is given by:

$$E_n = \hat{Y}_n - Y_n \tag{2.36}$$

Subsequently, the mean error ME can be calculated as follows:

$$ME = \frac{1}{N}\sum_{n=1}^{N} E_n = \frac{1}{N}\sum_{n=1}^{N}(\hat{Y}_n - Y_n) \tag{2.37}$$

It may happen that the ME is low, when the positive and negative errors cancel each other out. Therefore, the mean absolute error MAE is conducted, given by:

$$MAE = \frac{1}{N}\sum_{n=1}^{N} |E_n| = \frac{1}{N}\sum_{n=1}^{N} |\hat{Y}_n - Y_n| \tag{2.38}$$

The MAE values each error equally, independent of the size of the error. The fraction of ME and MAE is a measure of how much the bias is affecting the accuracy of the estimation. It is defined as follows:

$$\delta(MAE) = \frac{ME}{MAE} \tag{2.39}$$

Thus, $\delta(MAE)$ comes along with a value range of 0% to 100%, indicating the fraction of error which can be explained by the bias of the estimation.

Since it is also of interest to analyze how prone the estimation is to outliers, the *root mean squared error* $RMSE$ is obtained, given by:

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N} E_n^2} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(\hat{Y}_n - Y_n)^2} \tag{2.40}$$

To analyze the predictive capability of a model, the positional parameters ($\bar{Y}$, $\widetilde{Y}$, $\widehat{\sigma}_{\bar{Y}}$ and σ_Y) both of the target values and the estimations, as well as the error measures (ME, MAE and $RMSE$) of all estimations are compared for each conducted experiment. Note that for the decision-making, the MAE will be mostly used. Compared to the $RMSE$, the MAE is easier to interpret, and less prone to outliers.

2.4.2 Definitions of Statistical Hypothesis Tests

We use two tests to validate the significance of our results. First, we use the **unpaired two-sample t-test** to estimate the significance of deviations in means ($\bar{Y}$). Second, we use the **one-sided t-test** to estimate the significance of deviations in mean errors (ME, MAE and $RMSE$). Both tests as well as the underlying hypotheses are presented in the following.

Both used tests are t-tests, also known as Student´s t-tests, which is a group of statistical hypothesis tests, in which the test statistic follows a Student´s t-distribution for the null hypothesis. This distribution was developed by Student (1908) and arises when the mean of normally distributed populations with unknown standard deviations are analyzed. With a defined degree of freedom ν and n observations, which describes the number of values in a statistic that are free to vary, the t-distribution is defined as the sample mean relative to the true mean, divided by the sample standard deviation and multiplied by the standardizing term

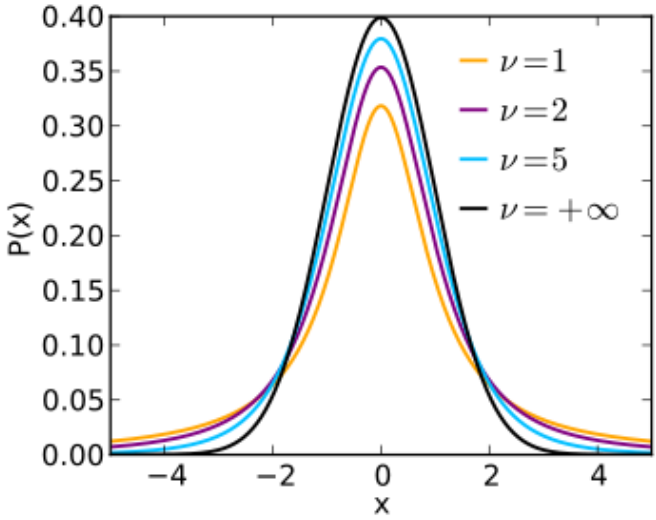


Figure 2.7: Student´s t-distribution under different degrees of freedom ν. (Wikipedia (2010))

$\sqrt{n}$. The t-distribution is visualized in Figure 2.7 for selected ν. There are two types of t-tests. First, the one-sample t-test, which is used when a sample mean is compared to a known distribution (e.g. normal distribution). Second, the two-sample t-test, which is used to compare two samples. If those samples are independent, meaning they are not drawn from the same distribution, the t-test is called "unpaired". Since both used tests are unpaired, we focus on this group in the following.

Given two samples (1, 2), this test is only applicable when the sample sizes are equal, both samples are normally distributed and the sample distributions share the same variance. While we will not violate the first two assumptions, we can not necessarily assume that the variances of both samples are equal. In this case, **Welch´s t-test** is applicable, because it does not assume equal variances (Welch (1947)).

This test can be used for two different kinds of relations. First, it can test for equality of the means. Using this variation, which is called **two-sided t-test**, we test for the following null and alternative hypothesis:

$\mathbf{H_0}$: The two population means $\bar{Y}_1$ and $\bar{Y}_2$ are equal ($\bar{Y}_1 = \bar{Y}_2$).

$\mathbf{H_1}$: The two population means $\bar{Y}_1$ and $\bar{Y}_2$ are unequal ($\bar{Y}_1 \neq \bar{Y}_2$).

Second, the test can be applied to test whether a one population mean is significantly higher than another population mean. It is called a **one-sided t-test**. Then, the null and alternative hypothesis look as follows:

$\mathbf{H_0}$: The population mean $\bar{Y}_1$ is greater than or equal to $\bar{Y}_2$ ($\bar{Y}_1 \geq \bar{Y}_2$).

$\mathbf{H_1}$: The population mean $\bar{Y}_1$ is smaller than $\bar{Y}_2$ ($\bar{Y}_1 \leq \bar{Y}_2$).

Regardless of the variation, the t statistic is then defined as follows:

$$t = \frac{\Delta\bar{Y}}{\sigma_{\Delta\bar{Y}}} = \frac{\bar{Y}_1 - \bar{Y}_2}{\sqrt{\sigma^2_{\bar{Y}_1} + \sigma^2_{\bar{Y}_2}}} \tag{2.41}$$

The degrees of freedom ν in the case of equal sample sizes N can be approximated by:

$$\nu \approx \frac{\sigma^4_{\Delta\bar{Y}}}{\nu_1^{-1}\sigma^4_{\bar{Y}_1} + \nu_2^{-1}\sigma^4_{\bar{Y}_2}} \tag{2.42}$$

with $\nu_i = N - 1$. To apply the test, we define a level of significance α, which represents the maximum probability that the null hypothesis is falsely neglected. In combination with ν, a p-value p is calculated, which represents the probability that, if the null hypothesis is correct, test results at least as extreme as the observed results are obtained. This value is then compared with the predefined α. If $p < \alpha$, the null hypothesis can be neglected.

In the analysis, we use both the two-sided, as well as the one-sided version of Welsh´s t-test. We use the two-sided version to estimate whether our estimations are significantly biased, by comparing the mean of the estimations and the mean of the actual cycle times. We use the one-sided t-test to estimate whether the

estimation error of one configuration of our methodology is significantly lower than from other configurations, or even from other, well-established approaches that are commonly used in the industry.

3 Literature review

In the following, recent efforts in the four areas of cycle time estimation in SC manufacturing are presented. An additional and extensive literature overview of waiting time estimation and feature selection in SC manufacturing is presented in Section 6.1 in the course of the publication presented in the next chapter.

3.1 Analytical modeling

Shanthikumar et al. (2007) have presented a survey on queueing theory for SC manufacturing systems and came to the conclusion that the practical use of those models is limited. Furmans et al. (2017) reviewed publications making use of queueing models. Baumann (2020) has developed a methodology to evaluate different service rules in a multi-queue environment. Nevertheless, Schlosser (2020) showed that models of queueing theory cannot be sufficiently applied to the real world environment in a SC fab, because those models are not sufficiently capable of handling the heterogeneous production environment. Especially on equipments where batches are processed, queueing models tend to estimate the waiting time distributions falsely. There is no sufficient queueing model to cope with all possible processing situations in a SC fab.

3.2 Simulation

Scholl and Domaschke (2000) have applied discrete-event-simulation to investigate the current situation in a fab of Infineon Technologies, leading to recommendations on actions how to reduce the impact of time constraints. Sivakumar and Chong (2001) have investigated the relationship between selected input and output variables in the backend of a SC manufacturing system with a data-driven discrete-event-simulation. Can and Heavey (2016) combine discrete-event-simulation with genetic programming to estimate cycle times of a production line. All those studies have in common that they show great performance in estimating cycle times in SC manufacturing, but they indicate in the Discussion the immense effort to build and maintain such models.

Because of this property of simulations, to estimate accurately at the cost of high building, maintenance and calculation time effort, they are also commonly used to validate other techniques. For example Veeger et al. (2010a) validated their system with a discrete-event simulation.

To summarize, simulation is the method of choice to validate models, because of its potential accuracy, while its building, maintenance and calculation time properties make them difficult for generalized cycle time estimation models.

3.3 Statistical Analysis

Pearn et al. (2009) modeled waiting times for single operations of each product type in SC packaging factories, using a gamma distribution. Subsequently, they combined those models into a cycle time estimation model. Akhavan-Tabatabaei et al. (2009) created a flow analysis model to forecast the WIP using existing correlations in arrival and service processes. This forecast can then be used in cycle time estimations.

Chen and Wang (2010) have applied a fuzzy c-means to classify jobs in a factory, which they used to construct a back propagation network (BPN) to estimate a cycle

time range over each confidence interval for each job category. This approach is problematic, since a confidence interval does not necessarily contain the actual value. Therefore, Chen and Lin (2011) randomized the parameters of a BPN to create a fuzzy cycle time forecast. Wang and Zhang (2016c) have created a cycle time estimation model based on the fuzzy c-means classifier in combination with a BPN by implementing a feature selection mechanism. They showed the positive correlation between data availability and solution quality. However, the estimation range was again based on confidence intervals, which do not necessarily contain the actual value. Wang et al. (2021) introduced a fuzzified BPN approach with threshold optimization for all input and output nodes to estimate cycle time ranges.

3.4 Hybrid Methods

While this field is comparably young, the amount of methods which combine analytical, statistical and simulation for cycle time estimations is constantly growing. Schelasin (2011) calculated factory variability on historical data by making use of a backward calculation algorithm, which he combined with the Kingman equation, and later improved it by a G/G/m formula (Schelasin (2013)). Veeger et al. (2010b) developed a simulation model to estimate mean cycle times of workstations for different scenarios in terms of throughput and product mix. Additionally, they have used a curve fitting procedure, which allows them to sufficiently estimate distributions under limited data availability for arrival and departure processes, as it is often the case in SC manufacturing (Veeger et al. (2010a)). Can and Heavey (2016) made use of genetic programming in combination with a discrete event simulation to estimate cycle times of a production line. Then, a machine learning algorithm was trained on the simulation results to imitate the whole production system. Yang et al. (2011) have combined queueing theory and simulation-based metamodelling to estimate mean cycle times for different workstations and product mixes. Additionally, the authors have developed another fitting approach based on neural networks (Yang et al. (2011)). A more general approach was developed by Hsieh et al. (2014), who provided a simulation metamodel, which is able to

characterize the relationship between the input and the target variable. This was then used to calculate the mean cycle time depending on the percentage of urgent lots.

A ready-to-implement solution combining queueing theory, simulation and statistical analysis was developed by IBM and is called "Enterprise Production Planning and Optimization System" (EPOS) (Brown et al. (2010)). It is a simulation system based on queueing theory, which can be integrated in the fab MES to capture routes, tools, raw process times, rework rates and WIP (Zisgen et al. (2008)). This tool can be also used to optimize production planning and management. In this context, tools are modelled as G^X/G(a, b)/c server queues and the manufacturing system as an open queueing network. Queue lengths for each equipment are determined with a decomposition approach in the open queueing network.

4 Semiconductor production process

For the production of SCs, silicon wafers with a thickness of about 1 mm are used as the starting product in the front end. Depending on the size of the chips, one wafer can contain several thousand chips. In SC production at Bosch in Reutlingen, wafers with diameters of 150 mm and 200 mm are processed. Up to 25 wafers are combined into one batch and transported together through the production line, which is called a fab. This transport takes place in special plastic boxes, which vary according to the size and type of wafers. As depicted in

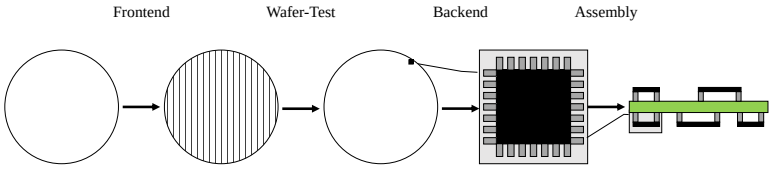


Figure 4.1: The four phases of semiconductor production (Klemmt (2012)).

Figure 4.1, the SC production is divided into four phases, the front-end, wafer test, back-end, and assembly. In the front-end, wafer processing takes place in a level A clean room environment (cle (2021)), since even the smallest contamination can damage the wafers. Here, several layers of material are applied to the wafer to produce micro-electric circuits. (Klemmt 2012, p. 5f)

In the subsequent wafer test, each silicon wafer is checked for damaged chips and sorted out if necessary. The wafer is then transported to the back end. There, it is shredded to separate the chips and assemble them into a package. Finally, SCs are combined into components in assembly. This is often done on a printed circuit board. (Klemmt 2012, p. 11-15)

4.1 The Frontend Production Process

Within this thesis the Frontend production process is of major interest since the majority of production steps are executed here and its complexity is significantly higher than in the other phases. The production process can be described as follows: Layers are brought onto the wafers by executing certain process steps repetitively. Hence, as depicted in Figure 4.2, SC production facilities are structured in work centers. According to Klemmt (2012), the Frontend production can be distinguished into six areas:

- **Lithography:** In lithography, a light-sensitive photo-resist is first applied through so-called coaters, and then the wafer is exposed through a mask. Depending on the product, this exposure can be carried out several times in succession. Finally, the wafer is developed to remove the unwanted areas of the photo-resist layer. Since Lithography is a very capital-intensive process and there are correspondingly few machines available, it often represents the production bottleneck. Additionally, Lithography is always the first step when carrying out a new layer on the wafer, which makes the throughput in Lithography a valid counter for the so-called mask-out (produced layers).

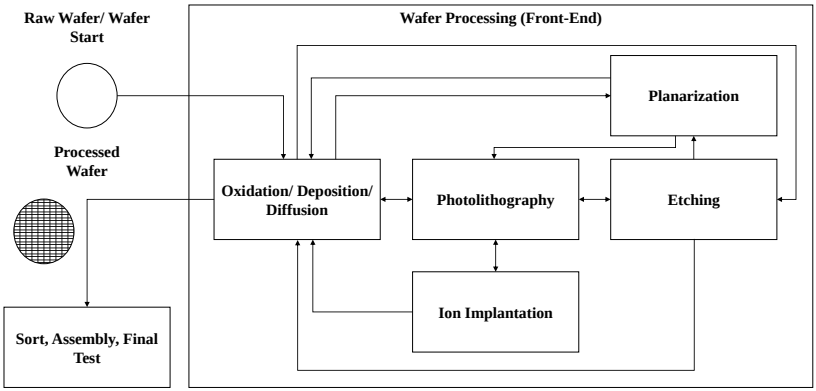


Figure 4.2: Work center production process in Wafer Processing (Mönch et al. (2011))

- **Doping:** Foreign atoms are deposited on the wafer surface to change the conductivity. This step can be done either by **Ion Implantation** or **Diffusion**. If Ion Implantation is used, ions are directed onto the wafer at high velocity within an electric field. The Diffusion process takes place at high temperatures in a furnace, causing the materials in the carrier gas to penetrate the SC crystal. Furnace processes in the diffusion area are often batch machines and lead to challenges in scheduling.
- **Oxidation:** The oxidation process takes place in a thermal furnace (1000 °C) under a pure oxygen atmosphere, where the surface of the Si-wafer reacts with the oxygen to a SiO_2-layer (oxid layer). The layer is necessary to protect and insulate the wafer. Like diffusion furnaces, oxidation furnaces are often batch machines and complicate scheduling.
- **Depostion:** To deposit layers of materials, either chemical vapor deposition (CVD) or physical vapor deposition (PVD) is used.

- **Etching:** In contrast to deposition, Etching involves the removal of layers. For this purpose, anisotropic (direction-dependent) dry etching or isotropic (direction-dependent) wet chemical etching can be used.
- **Measuring, cleaning and polishing:** Between the production stations, there are a large number of preparatory and inspection steps, such as cleaning steps, chemical-mechanical polishing or layer thickness measurement. For example, scrubbers remove tungsten and oxide by polishing the wafers. Apart from the regular inclusion of these steps in the production process at some points, they can occur also everywhere else in the process, due to regular cleansing steps of an equipment, or due to quality parameter measurements.

Depending on the type of product, these production steps are run several times by each batch in a different sequence. This results in a reentrant product flow, with batches at different stages of completion competing for the same machine capacity. Since machines often have to be retooled and adapted depending on the type of product, the work involved in scheduling becomes much more intensive. Complexity also increases as development engineers design and test new products in the same fab where regular wafer production takes place. In addition, so-called test wafers are used to check the parameters of the manufacturing process for correctness. These factors make SC manufacturing much more demanding in terms of scheduling and sequencing (Uzsoy et al. (1992)).

Routing in a SC wafer fabrication process is a complex iterative process. As depicted in Figure 4.3, the production process can be separated in non-repetitive stages. Within stages, a set of operations is executed, while the exact sequence can differ. Three reasons lead to this fact:

- **Rework:** Every operation has a certain probability to be executed incorrectly. Quality parameters are regularly measured and if they are out of a defined scope, the operation is defined as not successfully executed, which does not necessarily produce scrapping of an entire wafer, but results in rework. This rework could either be to repeat the operation, or to execute

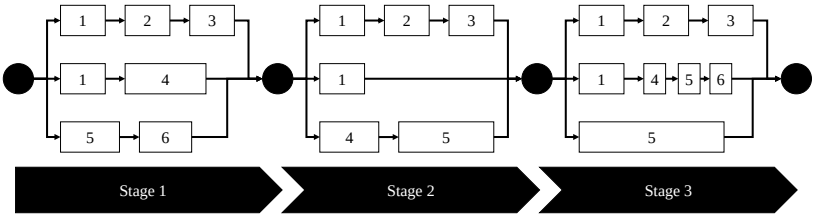


Figure 4.3: Route process through stages.

another sequence of operations (including removal of applied materials, cleaning steps and manual test steps).

- **Metrology:** There are operations which must be executed on each machine after a certain amount of operation executions or a certain amount of time. Hence, those operations are included in a fraction of operation sequences.
- **Process changes:** The operation sequence that will be executed in stages and the rework is changing regularly, mostly due to engineering-based process improvements or adjustments of the equipments. This unstable environment makes it difficult to interpret routes from the past.

Thus, an estimation of which route a lot will take is a non-trivial task.

While the industry is aware of the complexity, several techniques are commonly used to create cycle time estimations. We explain these techniques in the following, so that they can be used for a comparative analysis with our developed methodology in Chapter 9.

4.2 Industrial State-of-the-Art

The industrial state of the are regarding completion date estimation can be distinguished by their time horizon.

In the short-term – which typically means not more than a few weeks – discrete-event simulations are used. Because all processes can be realistically covered with this technique, it can be – in theory – completely accurate. Nonetheless, it comes along with heavy investments in the creation phase as well as in maintenance, as it intends to cover the production process as completely as possible. Since this process is undergoing constant change, this is a difficult task. Additionally, as already mentioned, the execution of discrete-event simulations are calculation-intensive – especially in complex production environments like SC fabs. Therefore, it is not typically used for long-term estimations, while it is commonly used to estimate completion dates of already started lots that are close to completion.

Long-term estimations – meaning the estimation of cycle times before the lots have entered the fab – are regularly made with mean estimators. The easiest approach here is to estimate a cycle time $ct_{l|p}$ of lot l and product p with the product's mean cycle time of all previous lots L:

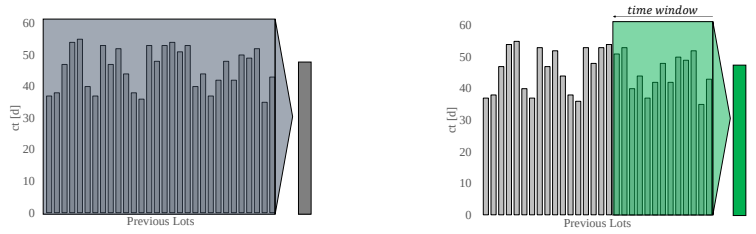
$$ct_{l|p} = \frac{1}{L} \sum_{i=1}^{L} ct_{i|p} \tag{4.1}$$

This approach is depicted in Figure 4.4 (a) and ignores the current fab situation, resulting in a complete insensitivity to any cycle time or fab development.

A more elaborated method, which is commonly used, estimates the cycle time of a lot with the average cycle time for all lots L in a defined time window δ:

$$ct_{l|p,t} = \frac{1}{L(t-\delta \rightarrow t)} \sum_{i \in L(t-\delta \rightarrow t)} ct_{i|p,t} \tag{4.2}$$

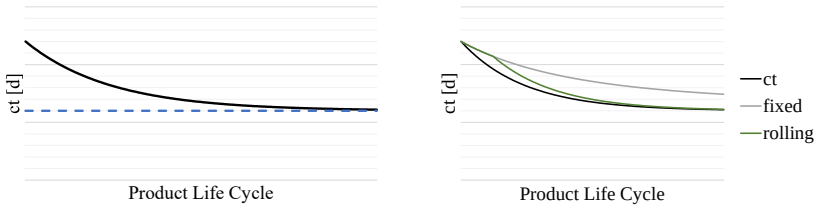
This approach is depicted in Figure 4.4 (b).



(a) **(b)**

Figure 4.4: (a): Mean estimations of cycle time. (b): Rolling mean estimations of cycle time.

An approach of rolling means is expected to work better for typical cycle time developments of products over their lifetime. A typical cycle time development over a product life cycle is depicted in Figure 4.5 part (a). As visible, the cycle time initially decreases rapidly, mostly due to engineering- and production-process-related improvements. Then, it saturates against a certain threshold of cycle time, which represents the high running behavior of this product.



(a) **(b)**

Figure 4.5: (a): Expected development of a product cycle time over its life cycle. (b): Expected behavior of both presented industrial standard approaches on the expected cycle time development.

The expected behavior of those two industrial standard approaches is depicted in part (b) of Figure 4.5. As shown, the fixed mean approach slightly decreases over time, but as it still considers each lot from the past, its decrease is slower than the

actual decrease, leading to a systemic error. The rolling average approach is more flexible, as old lots from outside its time window are not considered anymore. Hence, its decrease will sufficiently cope with the actual cycle time decrease after a short initialization period.

Both methods have in common that they do not need any maintenance, nor do they come along with heavy calculation complexity. Hence, they are easy to use, even for long-term situations.

It must be noted that there are also more advanced methods available to estimate the cycle time. For example an estimation based on one determining variable, such as the WIP, could be made. While this is slightly more flexible than the mean guessing method, it is still not capable of covering the large amount of inter-dependencies present in a SC production process. Since an inclusion of those methods would open up several degrees of freedom, we will not use them in this thesis.

Since this thesis aims to develop a methodology for long-term situations, both the mean as well as the rolling mean method are used as a comparison for the developed methodology in Section 9.6.5. Discrete-event simulation is only compared qualitatively, since the additional setup of a comparable simulation exceeds the scope of this thesis.

5 A concatenated cycle time estimation methodology

After the introduction on both the semiconductor manufacturing process as well as recent efforts in estimating cycle times, we present our own approach in the following.

The flow described in the previous chapter can be distinguished by so-called **stages** (s), which are sequential and non-repetitive by definition. Typically, the execution of a predefined series of stages sums up to a layer. Therefore, the cycle time of a lot (ct_l) can be described as the sum of the sojourn times (st_s) over all necessary stages S:

$$ct_l = \sum_{s=1}^{S} st_s \tag{5.1}$$

A stage consists of a sequence of operations o. This sequence is called a route (r). The sojourn time of any possible route (st_r) can be expressed as the sum of sojourn times over all operations of the route (st_o):

$$st_r = \sum_{o=1}^{r(O)} st_o \tag{5.2}$$

The route can differ, due to several reasons, which we will discuss in Chapter 7. Each operation sojourn time consists of a transportation time (tt_o), a waiting time (wt_o) and a processing time (pt_o):

$$st_o = tt_o + wt_o + pt_o \tag{5.3}$$

In the given case, transportation times are included in the waiting time, which means that the sojourn time of an operation can be expressed as follows:

$$st_o = wt_o + pt_o \tag{5.4}$$

Subsequently, the cycle time equals the sum of the waiting and processing times along the route of operations over all stages:

$$ct_l = \sum_{s=1}^{S} \sum_{o=1}^{O(r)} wt_o + pt_o \tag{5.5}$$

The share of the waiting and processing time on the sojourn time is depicted in Figure 5.1 (a). It is visible that the biggest part of the sojourn time is the waiting time. Additionally, the variability of waiting and processing time is depicted in Figure 5.1 (b). It becomes clear that also the major source of variability is the waiting time.

This assumption is well accepted in the community and can be explained with the autarky of the production process, leaving nearly no room for human interference. Therefore, it is common practice to assume fixed processing times through the technical processing times of the equipment. As this methodology is data-driven, we estimate processing times by their mean of past N executions:

$$pt_o = \frac{1}{N} \sum_{n=1}^{N} pt_n = \overline{pt} \tag{5.6}$$

We verify this statement as shown in Figure 5.2, since the processing time neither incorporates an underlying trend nor has great variance.

Fixing the processing time leads to three remaining variables in the equation, for which we will present strategies to approximate them in the following chapters:

- wt_o: We will identify a possibility to estimate the waiting time of an operation, and features that determine the waiting time in Chapter 6.

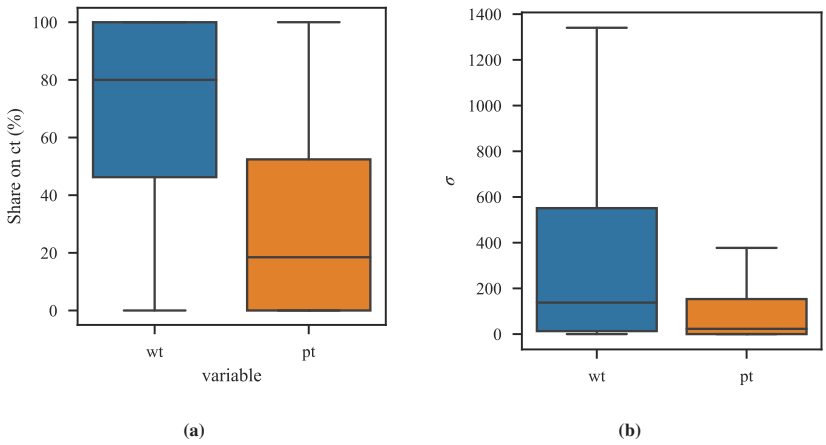


Figure 5.1: (a): Distribution of the share of the waiting (wt) and the processing time (pt) on the sojourn time. (b): Boxplot of the standard deviation (σ) of the waiting time (wt) and processing time (pt). The upper bar indicates the 95% percentile, the lower bar the 5% percentile. The upper limit of the box indicates the 75% percentile, the lower limit the 25% percentile. The line within the box indicates the median. Note that the data set which is introduced in Section 9.2 is used here.

- r: We will develop a methodology to calculate sojourn times over a selection of routes and weigh them with their probability in Chapter 7.
- wt_o *under uncertainty*: While all influencing features for wt_o can be measured at t_0, they are uncertain for future estimations. To cope with that problem, we will present a methodology to extrapolate the values of the feature set in the future in Chapter 8.

The complete methodology with pseudo-code for the final script is depicted in Figure 5.4 and works as follows. From the traces, we extract three sets of data. The first two are used to train and optimize models for each operation with get_model, as explained in Chapter 6. We use these models in $exec\_prediction$ to estimate the waiting and processing time of an operation. Additionally, we extract the routes with get_route and the feature set with $get\_features$ and return them based on the chosen method. In the combining method $get\_cycletime$ we use

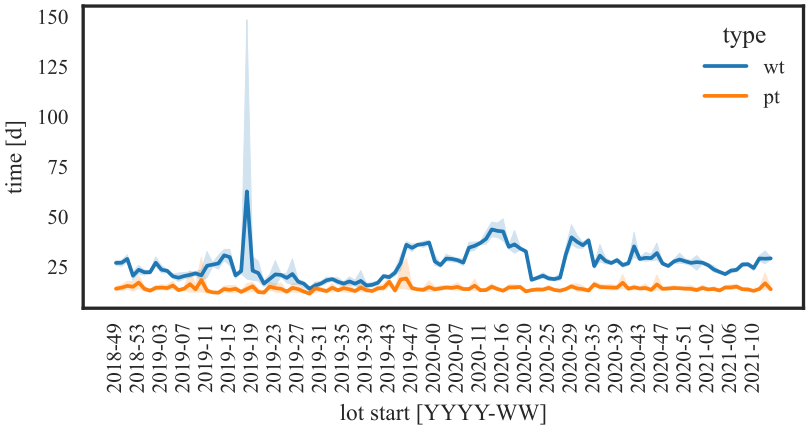


Figure 5.2: Mean total waiting time (wt) and mean total processing time (pt) in days as a function of time in calendar weeks. For each calendar week the means are computed over the set of lots started that entered the fab in that week. Note that the data set which is introduced in Section 9.2 is used here.

all these inputs to calculate the cycle time for every method. It should be noted that this figure is only an overview and provides information about the general input and output, while the sub-steps explained in the referring chapters are not mentioned, for example, the exact model optimization steps or the route probability calculation. Those steps are still included in the methodology, but not visualized.

Figure 5.3 shows the cycle time estimation from a process perspective. A route of N operations is chosen based on a route estimation policy. This refers to the route algorithm introduced in Chapter 7. For each operation o_n in the route, a model $M(o_n)$ is executed with a set of features, which is estimated based on a feature estimation policy. This mirrors the feature estimation options that we introduce in Chapter 8. We use this feature set in the model to estimate the operation‘s sojourn time $st(o_n)$. Subsequently, the cycle time ct is the sum of all operation sojourn time. Hence, the different methods that we explain in Chapter 6 and Chapter 7

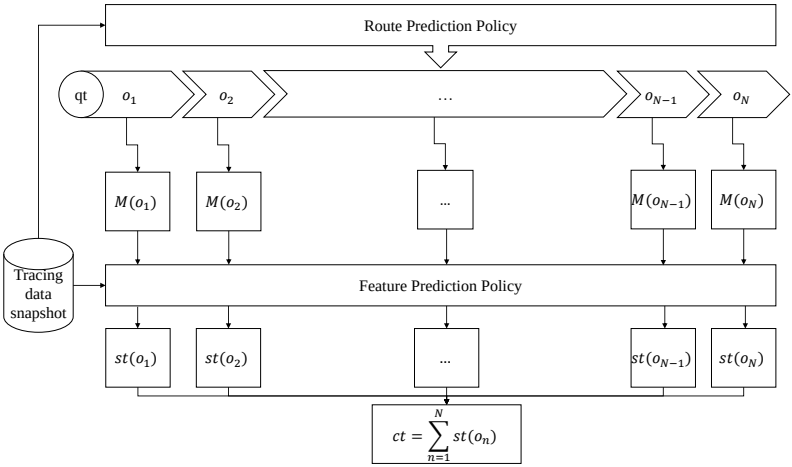


Figure 5.3: The proposed cycle time estimation approach

can be a source for several degrees of freedom for the methodology. Based on this, we present exploratory analyses in Chapter 9.

The expected advantages of the proposed methodology can be distinguished in three aspects. First, it is expected to react better to strong variation of cycle times in a SC fab compared with mean approximators currently in use for long-term estimations. Second, it is expected to be easier to build and maintain than discrete-event simulations, since no inter-dependencies have to be modeled for this methodology, because it is based on historical data. Because of its construction, with a concatenation of models along a route, process changes can be easily accustomed, since only affected parts of the system need to be modified. Third, its computation time is expected to be significantly shorter than a simulation, because no inter-dependencies of lots have to be considered for calculating a lot‘s cycle time.

Additionally, the methodology intends to be applicable to all sorts of production configurations. To apply it in other production environments, possibly relevant input features have to be identified and calculated. Furthermore, data pipelines

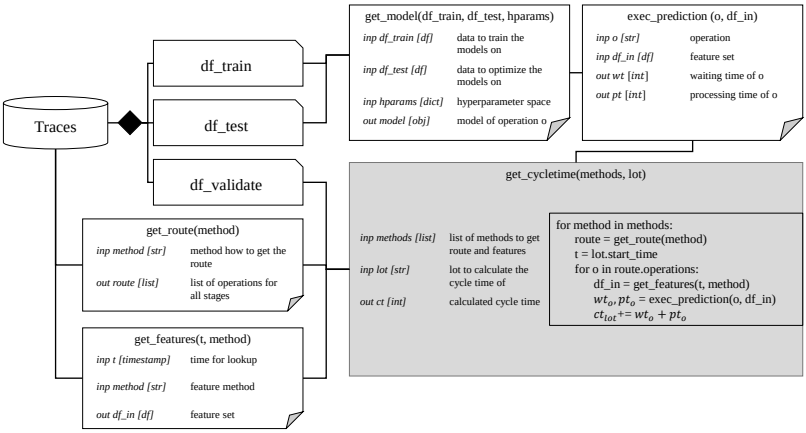


Figure 5.4: Overview of the complete methodology without sub-classes

have to be created, to feed the methodology. Apart from that, no adjustment to the methodology has to be made.

6 Creation of models for operational waiting time prediction

As mentioned in Chapter 5, the proposed methodology consists of a concatenation of models for operational waiting time estimation. Hence, we describe the creation of those models in the following. It consists of three steps: First, we identify a basic feature set, which is used for an initial training of the models. Second, we reduce the feature set, to tune the model performance. Third, we analyze patterns in the feature usage to create a study on the importance of certain features.

The chapter is based on Schelthoff et al. (2022)[1].
As the first author of this study the publishing house IEEE grants me the right to include this work in this dissertation.
The following text is taken from that paper without changes.

[1] **Authors contributions**
Conceptualization: Kai Schelthoff, Michel Janus, Kai Furmans
Methodology: Kai Schelthoff, Michel Janus, Eva Schlosser, David Plohmann
Design of Experiments: Kai Schelthoff, Michel Janus, Christoph Jacobi
Formal analysis and investigation: Kai Schelthoff, Michel Janus, Christoph Jacobi, Eva Schlosser, David Plohmann
Writing – original draft preparation: Kai Schelthoff, Christoph Jacobi
Writing – review and editing: Kai Schelthoff, Christoph Jacobi, Michel Janus, Kai Furmans

SC manufacturers are faced with increasing customer requirements regarding demand, functionality, quality, and delivery reliability of microchips. This constantly growing market pressure necessitates accurate and precise performance estimation for decision-makers to enter delivery commitments with customers. One significant performance measure is waiting time, which frequently accounts for the highest proportion of cycle time and contributes the most to its variance. While there are many studies which predict cycle time, we prefer to address waiting time as the variable of interest and allow the practitioner to decide how they want to estimate processing times (i.e. deterministic or stochastic). To obtain the accumulated total waiting time in a semiconductor fab, one could conduct individual predictions for each operation and sum them up for the entire production cycle of a lot. Predicting waiting times is, however, a non-trivial task since numerous potentially important influencing features must be considered. Prediction models that consider a great variety of features are computationally extensive and prone to over-fitting while, in contrast, basic models fail to provide valuable predictions. Consequently, semiconductor manufacturers are confronted with the task of identifying the relevant feature set for waiting time prediction.

Furthermore, European semiconductor manufacturers are confronted with a volatile demand for a plethora of products. Consequently, semiconductors are produced in so-called *high product-mix / low-volume* (HMLV) semiconductor wafer factories. The competitive advantages of HMLV manufacturers are flexibility (managing a plethora of products), and agility (adapting to new constraints rapidly without threatening costs) at the cost of significant increase of interdependencies. In a HMLV wafer fab, the product mix, available technologies, and production capacities constantly evolve over time and a multitude of operations are processed simultaneously on heterogeneous tool sets. This complex production environment implies a multitude of additional features correlated with the waiting time, but so far it is unclear how these features contribute to the forecast quality.

To address this problem, we investigate a plethora of features for waiting time estimation based on real operational data from the Robert Bosch GmbH. We propose a method to predict single waiting times per lot and operation at the

point of the completion of the previous operation. We demonstrate the method with real operational data from two production areas, namely Lithography and Diffusion. To the best of our knowledge, we are the first to attempt for waiting time estimation in a HMLV wafer fab.

This paper is organized as follows: Section 6.1 presents the related literature. In Section 6.2, we introduce the methodological approach and define the feature set. In Section 6.3, the framework is applied on a data set from an HMLV fab in Germany belonging to the Robert Bosch GmbH and our results are presented. We discuss the results achieved in Section 6.4 and derive managerial implications in Section 6.5. Section 6.6 draws conclusions on our study and identifies further research needs.

6.1 Literature Review

The literature categorizes the approaches on estimating cycle and waiting times in semiconductor wafer fabs into five areas. The first area is analytical models with queueing theory being one main example. Queuing theory models allow for fast computation of key performance indicators based on the stochastic analysis of arrival and service processes (see Shanthikumar et al. (2007) for an extensive review, and e.g. Schelasin (2011), Akhavan-Tabatabaei et al. (2009), Zisgen et al. (2008), Morrison and Martin (2007) for recent approaches). Second, extensive simulation studies (Dilefeld et al. (2020), Seidel et al. (2019), Yang et al. (2008), Johnson et al. (2005)) are conducted for performance evaluation of semiconductor wafer fabs. However, building statistically significant simulation models is time-consuming and increasingly harder for complex manufacturing systems. Both approaches seem to be unsuitable for our use case and are therefore not discussed in more detail.

The third area is statistical approaches, which include probability-based and regression-based models. Backus et al. (2006) conducted cycle time prediction and policy control based on production line status and data mining. Tai et al.

(2012) calculate the cycle time for final testing with the assumption of Weibull-distributed waiting times. Hassoun (2013) shows that segmentation of the process line leads to improvements in the predictability of the forecast of cycle time. Wang and Zhang (2016b) use big data analytics to predict cycle time and correlation analysis for feature selection (Wang et al. (2018b, 2020)) of suitable variables.

The fourth area is artificial intelligence models. A post-classification approach with an artificial neural network is proposed by Chen et al. (2008) to cluster jobs with similar cycle time forecast accuracy. Chen et al. (2009a) consider lot cycle time prediction during production ramp-up using a self-organization map fuzzy-back-propagation network algorithm. A fuzzy-neural approach for estimating the remaining cycle time of each job is proposed in Chen et al. (2009b). In recent publications, a bi-directional classifying fuzzy-neural approach is developed to aggregate the results of pre-classification and post-classification (Chen (2011)), and a procedure is developed to divide jobs into several groups according to their estimation errors (Chen (2016)). Meidan et al. (2011) present a detailed study to determine relevant influencing features by using simulated data. Tirkel (2013) compares the prediction quality of a Decision Tree model and a Neural Network model for cycle time prediction of a single operation step, a line segment and a complete production line. Lingitz et al. (2018) use real operational data to forecast lead times for the production steps Sorter, Bakefuse and Sputter, focusing on the performance of different machine learning algorithms. Zhang et al. (2018) propose an imperialist competitive algorithm incorporating remaining cycle prediction for photolithography machines' scheduling problem. Wang et al. (2018a) design a density peak based radial basis function network to forecast cycle time. Chakravorty and Nagarur (2020) present a cycle time prediction methodology based on a back-propagation trained artificial neural network which can be used for making real time dispatching decisions at trigger steps of queue time restricted zones.

The fifth and last area is hybrid models, which combine some of the aforementioned approaches. Chen (2007) uses real data collected from a semiconductor fab to predict the job completion time using fuzzy c-means and a back-propagation network ensemble. Chen and Wang (2010) extend this approach by incorporating

a nonlinear programming model and present and iterative procedure to deal with outliers (Chen and Wang (2013)).

The studies presented show that many researchers focus on improving forecasting models and designing more sophisticated neural network or data mining models to forecast cycle times. However, as Wang et al. (2018b) recognize, relatively few studies focus on feature selection to exploit the potential of highly correlated input data to predict the output value. This process is, however, crucial in our use case, since we are analyzing real operational data of a HMLV wafer fab. In contrast to former studies, which were mostly subject to some limitations (such as the analysis of selected tool groups, or the consideration of a limited number of product types), we analyze highly volatile production data for the production areas Lithography and Diffusion. In related studies, some global features are widely used as input values. We adopt these features, and introduce novel features based on the expert knowledge of line experts in our fab to incorporate the special behavior of the HMLV fab under consideration. Therefore, our set of potential key features increases, and feature selection becomes even more necessary to build lightweight prediction models.

6.2 Methodology

It is well known that cycle time is one of the most relevant performance measures for semiconductor manufacturing processes. Cycle time is defined as the elapsed time between starting and completing a task, which is composed of transport time, waiting time, processing time, and time for additional steps (Tirkel (2013)). The Manufacturing Executing System (MES) of our fab tracks Move-In and Move-Out times of each machine (that is, start and end of each processing step). After completing the previous task, the lots enter the joint waiting room of the tool group of the next processing step and wait to be processed. Note that the waiting room is not physically co-located to the tool group and upon arrival of a lot, it is not determined which machine will process the lot. Consequently, the waiting times in our study include transport times between the tool groups.

The dispatching strategy of the waiting room is dependent on various factors, not FIFO. In previous studies, processing times were assumed to be constant for a given processing step. However, in our use case, the processing times are found to be subject to some fluctuations. Nevertheless, the fluctuation of the waiting time outreaches the processing time‘s fluctuation by far. Therefore, in this study, our focus is on analyzing and forecasting the waiting times, while the behavior of the processing time in the past is used as an independent variable. We define the dependent variable of our models to be the expected waiting time per lot at a given tool group upon arrival at the tool group at t_0.

6.2.1 Feature investigation

Table 6.1 shows the feature set of our study. The feature set indicates the manifold of this feature, either for nominal categoricals (nominal cat.), which have to be one-hot-encoded, or for ordinal categoricals (ordinal cat.) and continuous (cont.), which are often times collections of features. The features stem from two sources. First, we investigate and adapt features based on past publications which are applicable to our use case. Second, with the help of line experts from the fab where the data stem from, we identify new features which have not yet been included in the literature.

6.2.1.1 Adapted features

The features used in literature are also depicted in Table 6.1. We are aware that some of the mentioned publications use more features than depicted, but we focus on those which are applicable to our case. Publications dealing with cycle time predictions or just one equipment can possibly use a different feature set. To use these features in our case, we have to partially adapt them. In the following, each feature is briefly explained, including why we assume an importance and adaption mechanics if necessary.

- **Lot priority** (P)**:** Each lot is assigned a priority at fab entry.

Table 6.1: Features considered in the analysis

Origin	Feature	Abbreviation
Literature	Priority	P
	WIP in fab	WIP
	WIP in queue (prod./non-prod.)*	wip_X
	Job Arrival Time	qt
	Inter-Arrival-Time	IA_{preX}
	Inter-Departure-Time	ID_{preX}
	Machine (group) utilization	u_{preX}
	Machine (group) availability	a
	Prev. (1/3/10) waiting time*	$min/max/\mu/\sigma^2(wt_{preX})$
	Prev. (1/3/10) processing time*	$min/max/\mu/\sigma^2(pt_{preX})$
	Nbr. of operation loops	l
	Product mix in fab	pm_{fab}
Experts	Product mix in queue	pm_{queue}
	Nbr. of different products in the queue	n_{queue}
	Time since last departure	dt
	WIP profile	WIP_{dist}
	Similar operations waiting	ql_{sim}
	Waiting times at t_0 in queue	$wt_{dist\|t_0}$
	Level of completion of lot at t_0	$compl_{t_0}$
	Shift	S
	Weekend	w
	Holidays	h
	Previous operation	o_{prev}
	Layer	L
	Stage	$St_{cur/total}$

* This feature represents a set of features which is summarized for clarity.

In all further steps, those features are handled independently.

Table 6.2: Type and set size of the considered features.

Abbreviation	Type	Set Size
P	Nominal cat.	3
WIP	Ordinal cat.	1
wip_X	Ordinal cat.	3
qt	Cont.	1
IA_{preX}	Cont.	2
ID_{preX}	Cont.	2
u_{preX}	Cont.	2
a	Ordinal cat.	5
$min/max/\mu/\sigma^2(wt_{preX})$	Cont.	9
$min/max/\mu/\sigma^2(pt_{preX})$	Cont.	9
l	Ordinal cat.	1
pm_{fab}	Cont.	10
pm_{queue}	Cont.	10
n_{queue}	Cont.	1
dt	Cont.	1
WIP_{dist}	Ordinal cat.	10
ql_{sim}	Ordinal cat.	1
$wt_{dist\|t_0}$	Cont.	10
$compl_{t_0}$	Cont.	1
S	Nominal cat.	3
w	Nominal cat.	2
h	Nominal cat.	2
o_{prev}	Nominal cat.	var.
L	Nominal cat.	var.
$St_{cur/total}$	Nominal cat.	var.

Table 6.3: Acknowledgment of features in past publications

Paper	Feature								
	P	WIP	wip_X	qt	IA	ID	u_{preX}	a	wt_{preX}
Chakravorty and Nagarur (2020)		✓							
Wang et al. (2020)	✓	✓	✓				✓		
Lingitz et al. (2018)	✓	✓	✓	✓	✓	✓	✓	✓	
Wang et al. (2018b)	✓	✓	✓				✓		
Zhang et al. (2018)	✓	✓	✓				✓		✓
Wang and Zhang (2016c)		✓					✓		✓
Chen (2016)		✓	✓				✓		✓
Meidan et al. (2011)		✓	✓				✓	✓	✓
Chen (2007)		✓	✓				✓		✓
Backus et al. (2006)		✓							
Chung and Huang (2002)				✓			✓	✓	✓

	pt_{preX}	$AMHS$	l	pm_fab
Chakravorty and Nagarur (2020)	✓			
Wang et al. (2020)	✓	✓		
Lingitz et al. (2018)	✓			
Wang et al. (2018b)	✓			
Zhang et al. (2018)	✓			
Wang and Zhang (2016c)				
Chen (2016)				
Meidan et al. (2011)	✓		✓	
Chen (2007)				
Backus et al. (2006)				
Chung and Huang (2002)	✓			✓

  - *Importance:* This priority refers to the importance and urgency of the lot, which is especially important for scheduling during manufacturing and therefore considered an influencing feature.

- **Work-in-progress (WIP):** The WIP is defined as the number of lots currently in operation in a machine group and the number of lots currently waiting in front of the machine group.

  - *Importance:* This feature is considered an important feature impacting the cycle time in nearly all publications dealing with this topic. This is in line with Little's law, which defines a direct correlation between the amount of customers in a system and their time in the system.

  - *Adaptation:* Since there exist productive and non-productive lots, i.e. lots used for testing and maintenance purposes, the WIP for all jobs are calculated for productive lot types (wip_p) and for non-productive lot types (wip_{np}) individually. The resulting total WIP in the machine group equals the sum of both features, but is not used as a feature to avoid redundant information. Additionally, the WIP of the total fab (WIP) is considered.

- **Arrival time in the day (qt):**

  - *Importance:* It is of relevance for batch-building (group of lots to be processed together) operations, in which rate other lots arrive or depart.

- **Inter-arrival (IA) and inter-departure times (ID):** Let at_l be the time of the arrival and dt_l the time of the departure of lot l. IA and ID are defined as the time between the arrival/departure of the current and the previous lot of the same operation type:

$$IA_l = at_l - at_{l-1} \tag{6.1}$$

$$ID_l = dt_{l-1} - dt_{l-2} \tag{6.2}$$

  The order of the lots is defined by the corresponding arrival timestamp.

  - *Importance:* For batch operations, it is of importance in which frequency other lots arrive.
  - *Adaptation:* For both features, the last inter-arrival (IA_{pre1}) and inter-departure time (ID_{pre1}) as well as the rolling average of the last 10 values ($IA_{pre10}; ID_{pre10}$) are utilized as features.

- **Utilization of machine groups (u):** For each machine m in machine group M there is an available processing time ($ca_{t|m}$) and an occupied time ($cu_{t|m}$) in a defined time window $t = t_0 - x \rightarrow t_0$, e.g. an hour. They can be expressed as follows, with M as group of machines capable to process o:

$$ca_{t|M} = \sum_{m \in M} ca_{t|m} \tag{6.3}$$

$$cu_{t|M} = \sum_{m \in M} cu_{t|m} \tag{6.4}$$

  The utilization (u_{preH}) is the share of the occupied time on the available processing time:

$$u_{preX} = \sum_{t=-X}^{t_0} \frac{cu_{t|M}}{ca_{t|M}} \tag{6.5}$$

  - *Importance:* The utilization of the equipments indicates the available capacity for the process execution and is therefore a common feature.
  - *Adaptation:* We obtain both, the utilization in the past hour (u_{preH}) as well as in the past day (u_{preD}) to indicate recent developments in the utilization of the equipments.

- **Availability of machines (a):** In the context of this paper, the availability is defined by the number of available machines which are able to execute the operation.

  - *Importance:* This is a common feature in literature, as the availability of processing tools has direct impact on the waiting time in the machine group queue, as it determines the availability of capacity.
  - *Adaptation:* We obtain the number of machines in each equipment state ("available", "repair", "maintenance", "setup", and "shutdown") as features in order to enable learning on the composition of the machine states in the machine group and its consequences on the waiting time.

- **Processing time (pt_{preX}) and waiting time (wt_{preX}):**
  - *Importance:* Several publications use the previous cycle time as influencing features to predict cycle times.
  - *Adaptation:* In this paper, we split up the cycle time to acknowledge the fact that both values do not share the same distribution. Additionally, we indicate both values of the last finished operation, of the previous 3 and of the previous 10 recently finished operations of the same product-operation-combination, because it could help to indicate recent trends in both values. Since these features vary (except for the very previous waiting and processing time), the minimal (min) and maximal (max) value, the mean (μ) and the variance (σ^2) of wt and pt are added as features.

- **Product mix in the fab (pm_{fab}):**
  - *Importance:* An increasingly complex product mix is more challenging and therefore further increases the planning complexity. Since increased complexity impacts the performance of dispatching algorithms, it can be used as an indicator of the stress level of production planning, in combination with the overall fab WIP.
  - *Adaptation:* The complexity of a product can be measured by the amount of layers necessary for its completion. Hence, we indicate the product mix by the deciles of layers necessary for the completion of all products in the

fab at arrival time, as well as of all lots in the queue of the equipments which are capable of executing the operation.

- **Number of tool loops (l):** This feature indicates whether an operation is executed for the first time, or is repeated as a rework step.
  - *Importance:* The underlying assumption is that a rework step could get urgent or could get extra attention from planners, since it is an unforeseen event.

6.2.1.2 Novel features

In the following, the novel features are listed, including why we assume them to be a relevant feature for waiting time estimations.

- **Product mix in the queue (pm_{queue}):** Despite of the aforementioned pm_{fab}, we conduct this feature using the same calculation pattern.
  - *Importance:* Similar to pm_{fab}, pm_{queue} is an indicator of the planning complexity of the machine group and may be of interest in highly sequence-dependent production areas, because it indicates the heterogeneity of a queue. Hence, it might be of relevance for waiting time estimations.
- **Number of different products in the queue (n_{queue}):**
  - *Importance:* It may be of importance in areas with sequence-dependent setup times, since a heavy variety of products may lead to increased setup times and therefore higher waiting time.
- **WIP profile (WIP_{dist}):** This feature is a measurement of the level of completion of all lots in the fab at t_0. It can be calculated as the fraction of completed layers and all necessary layers of a lot. We introduce the WIP profile as deciles for the whole fab as well as for lots in the queue of the machine group.

  - *Importance:* WIP profile is an essential feature because it is used to decide how the queues are steered. Products which are close to completion (that is, products which have a high WIP profile value) are likely to be preferred by the dispatching algorithm as its completion is directly influencing the output of the fab, which is a key performance metric.

- **Level of completion ($compl_{t_0}$):** This feature indicates the fraction of layers already completed and the total amount of layers of the lot we are currently predicting.

  - *Importance:* With this featuere we acknowledge the importance of the completion level not only for all concurring lots, but also for the lot to be predicted.

- **Amount of similar operations in the queue (ql_{sim}):**

  - *Importance:* Similar operations are of the same operation type (independent of its product) and can be therefore produced in batches, if the equipments are capable of processing batches. Hence, a lot could be preferred if a lot of similar operations is waiting for execution to create full batches. It is expected that this feature shows no positive effect on operations where no batch process is involved.

- **Waiting times of all lots waiting in the queue at t_0 ($wt_{dist|t_0}$):** In order to keep a fixed shape of input features, we group the waiting lots in the queue into deciles of waiting times.

  - *Importance:* This feature is used to extract further information about the queue participants.

- **Shift at t_0 (S):** Early: 6:00-14:00, late: 14:00-22:00 and night: 22:00-6:00. **Weekend at t_0 (w):** 1 if lot enters the queue on a weekend, else 0. **Holidays (h):** 1 if lot enters the queue during national holidays of the fab location, else 0.

  - *Importance:* We assume that personal resources differ between shifts, weekends and holidays.

- **Previous operation ID (o_{prev})**: This categorical feature is introduced, since in our use case, the transportation time is included in the waiting time.

  - *Importance:* We assume that it can work as an estimator for the distance to be transported within the fab.

- **Time span since the last departure of a product with the same operation (dt):** This feature indicates whether an operation is executed regularly, rarely or if the operation is new.

  - *Importance:* The underlying assumption is that the production efficiency is higher for high-runner products.

- **Layer (L)** and **stage (St_{cur}) of the current operation:** This feature indicates the lot's position in the fab.

  - *Importance:* These features might be of interest since products are treated differently when they are close to completion, or facing a capital-intensive stage or layer.

- **Number of total stages necessary (St_{total}) for completion:**

  - *Importance:* This feature shall indicate how complex the respective lot is, assuming that more complex products shall be of higher priority in certain dispatching situations.

6.2.2 Feature Selection Framework

The proposed feature selection process is composed of three steps which are executed for each product-operation-combination. The following methodology has

been derived from a combination of a permutation feature importance calculation (Breiman (2001)) and a sequential backwards search based on the permutation feature importance values (Huang et al. (2016)). The data set for each part-operation-combination is divided in a training (50%), a test (25%), and a validation set (25%) by a random split, applying scikit-learn train-test-split. In the setup phase of this study we compared the results using a random split with a time-dependent split. The results were comparable, but since the data set contains different value ranges over time, we decided to work with a random split.

For each product-operation-combination, we first train a random forest classifier using the scikit-learn library (Pedregosa et al. (2011)) with the training data set and execute hyper-parameter tuning using the test set. In the set up phase of this study, we also tried other modeling techniques (e.g. Multi layer Perceptrons, Recurrent Neural Networks), and the results show to be comparable. Hence, we decided to focus on Random Forests in the course of this study. We trained a model for each product operation-combination as the so-called baseline model, using all previously introduced features. Second, we evaluate the performance of the baseline model on the validation set in order to ensure that the model is evaluated on unseen data. Note that only baseline models with a sufficient performance score are suitable for feature selection and consequently, all models with low predicting capability are erased from further analysis. In the third step, a Permutation Feature Importance (PFI) based feature reduction is executed for each model. A scikit-learn based model with optimized hyper-parameters is trained with only the identified relevant features. Finally, we evaluate the performance of the optimized model of a given part-operation-combination against the corresponding baseline model on the validation set.

6.2.2.1 Baseline Model

The optimal set of hyper-parameters is chosen by a grid search with the python package scikit-optimize (sko (2020)). The boundaries of the grid search can be seen in Table 6.4. The hyper-parameters optimized are described in the following, all other parameters of the method are left at default values. A random forest

Table 6.4: The hyper-parameter space for all random forests

Hyper-parameter	Value Range
estimators	200 - 2000
max depth	10 - 110
max features	[auto, sqrt]
min samples split	[2, 5, 10]
min samples leaf	[1, 2, 4]
bootstrap	[True, False]
warm start	[True, False]

is built alongside various hyper-parameters. First, the number of **estimators** determines the number of decision trees within the random forest. Second, the **max depth** determines the maximum allowed depth of each decision tree. Third, the **max features** determines the number of features to consider when looking for the best split. If it is "auto", then the maximum features is the total number of features. If it is "sqrt", then the square root of the total number of features is chosen. The hyper-parameter **min samples split** determines the minimum number of samples required to split an internal node. The hyper-parameter **min samples leaf** determines the minimum number of samples required to build a leaf. Hence, splitting points are only considered to be implemented in the tree if it leaves the defined amount of training samples for the other branches. The hyper-parameter **bootstrap** defines whether bootstrap samples are used for building the trees. Finally, the hyper-parameter **warm start** defines whether the solution of the previous call is reused when building the forest, or if a whole new forest is fitted.

6.2.2.2 Baseline Model Evaluation

Since we face a regression problem, we evaluate the model performance based on the coefficient of determination (R^2). Let $\bar{y}$ denote the mean of n observations

$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$, and f_i the corresponding prediction of the random forest baseline model. R^2 is defined as one minus the share of the explained sum of squares (SS_{res}) in the total sum of squares (SS_{tot}):

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y})^2}. \tag{6.6}$$

Hence, R^2 for a given model is 1, if all estimates f_i equal the observations y_i, and 0, if all estimates equal the mean $\bar{y}$.

6.2.2.3 Permutation-based Feature Set Reduction

In the following, we execute a permutation feature importance algorithm Breiman (2001), which is then used as a sorter in a sequential backwards search. Starting with the described baseline model and its performance s, for each feature j, the values in the data set are randomly permuted K-times and the resulting model performance s_{kj} is computed. We deploy the aforementioned coefficient of determination R^2 as performance measure s_{kj}. The importance i_j of feature j is defined as the resulting decrease in the model performance by this shuffle

$$i_j = s - \frac{1}{K}\sum_{k=1}^{K} s_{kj}. \tag{6.7}$$

To reduce the influence of random fluctuations in PFI, this process was carried out $K = 1000$ times for each feature in every model. One could argue that lower values are also acceptable, but since this step is computationally inexpensive (since no models must be retrained), we chose to do it that often.

Afterwards, to identify the optimal feature set for a given problem, we use a sequential backward search as proposed by Huang et al. (2016), where the PFI is used as a sorter. This method was compared with a common method for estimating feature importance in random forests, namely the Gini Impurity feature

importance (Pedregosa et al. (2011)), in the set up phase of this study. We chose to instead use the proposed method due to the known problems in high-cardinality features within this alternative.

6.3 Framework Application

We apply the proposed methodology to real operational data. We first briefly describe the use case. Afterwards, we analyze the results when applying the methodology on the described use case.

6.3.1 Use Case Description

The features are evaluated on real operational data from a Bosch Wafer Fab in Germany. Within this study, we focus on the so-called Diffusion and Lithography work centers (for more information on the details of those work centers, see Mönch et al. (2011)). We do so, because they are either tending to be the most repetitive step in the production process (Lithography), or be capital intensive, which makes it important to keep utilization high, which subsequently results in higher waiting times. Additionally, those areas differ in their configuration. In Lithography, single lots are processed with relatively short processing times and therefore higher fluctuation in the queues. The loading is done completely automated. It is especially interesting to analyze Lithography since in the given case it comes along with significant sequence-dependent setup times. This is due to the fact that so-called reticles, which define the chip layout, are used. If two product-operation combinations in the sequence can reuse the same reticle, no setup time is needed. In opposite, Diffusion operations are produced in so-called batches, which means that groups of lots are produced together, with comparatively long processing times. This again could possibly increase the importance of some features. Additionally, the loading is executed manually, and no significant sequence-dependency is given. Hence, those two production areas are ideal to analyze the resulting differences in feature importance. Data from

the highest volume products from January 2017 until October 2020 was used, with over 14,000 lots and a total number of over 1.4 million operations located in Lithography and Diffusion. The highest volume products are of big interest for a precise prediction, as customers rely on this information in a Just-in-Time or Just-In-Sequence production. Within this study, we filter out operations that are either always very short (e.g. test and transportation steps) or occur very rarely (e.g. some rework steps). In both cases, the practical use of a model would be very limited. Hence, we have added a constraint of at least 1,000 data points and at least ten minutes of median waiting time. Additionally, since we are working with real operational data, invalid data points are in the data set. In the absence of an identification possibility, we only use the 99.9 percentile in overall cycle time of lots. Given the fact that each model will be trained on at least 1,000 data points, subsequently at least one data point per model is removed as an outlier. Even after removal of outliers, both our target feature (waiting time) as well as the predictiveness of our input features vary strongly.

6.3.2 Results

We trained models and optimized their hyper-parameters for 262 product-operation combinations, as described in Section 6.2.2.1. 199 of those models create predictions for Diffusion operations, while 63 relate to Lithography operations, as the variety of operations in this production area is smaller. As described in Section 6.2.2.2, we constrain our analysis to baseline models with a reasonable minimum of predictive power. Hence, before we perform the feature importance analysis, we have to reject models, which are not predicting with satisfactory accuracy. We assume a model to perform well in the given case, if R^2 is greater than 0.3. The threshold was manually chosen to retain as many models as possible while ensuring that an observed change in predictive accuracy can be safely distinguished from statistical fluctuations, but has to be discussed in further review (see Section 6.4). Applying this filter step to all our models erases 131 models from further analysis and leaves 131 models for the next step. For the remaining 131 models, we observe a R^2 value of 0.55 for Diffusion and 0.56 for Lithography

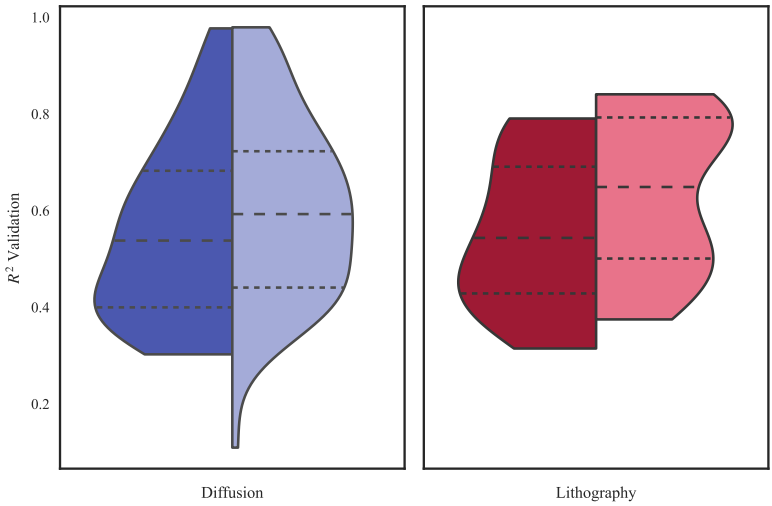


Figure 6.1: Violinplot of the R^2 distribution before (left) and after (right) the feature reduction by production area (dotted lines representing the quartiles)

on the validation data. For each of those models, the permutation feature importance was calculated and used as a sorter for the sequential backwards search as described in Section 6.2.2.3.

The performance of the methodology is depicted in Figure 6.1 and Figure 6.2. Figure 6.1 shows violinplots of the R^2-score of the baseline models (darker colors) and the reduced models (brighter colors) distinguished by production area. As visible, when using the reduced feature set for waiting time predictions, the model performance remains stable. The mean R^2 is only weakly affected (+0.054; +0.049 in Diffusion, +0.079 in Lithography). From all 131 models, 2 are pushed below the $R^2 = 0.3$ threshold (both in Diffusion).

To analyze the impact of the feature reduction in more detail, Figure 6.3 shows the median R^2-performance on the test set as a function of the size of the feature set. For each number of features, we report both the median and the 95% quantile

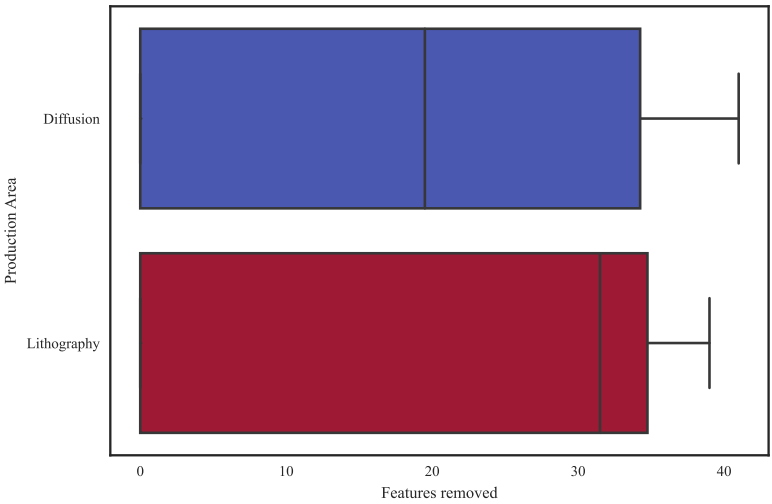


Figure 6.2: 95% Boxplots of the change in feature set size by production area

performance out of all models across product-operation-combinations and different feature sets. It can be seen that the R^2-score drops sharply when going below five features. This tells us that on average there are only around five features used to predict the waiting time. This threshold varies only slightly depending on the product-operation-combination. Interestingly, there is also a slight increase in the model performance when using 5-10 features, compared to models using all features. This shows one of the benefits of applying feature selection, as it safeguards the machine learning algorithm against an overabundance of features. While the performance of an ideal machine learning algorithm should not degrade when using more features, small degradations are possible due to distributing the same data set across more dimensions during training.

At the same time, the size of the feature set could be significantly reduced, as depicted in Figure 6.2. It can be observed that the approach proposed within this study reduces the feature set by over 22 features on average and over 20 on median

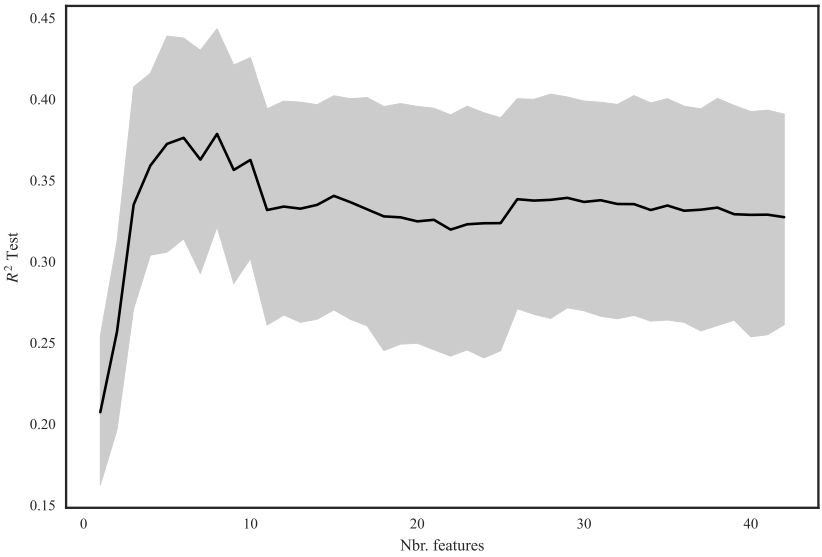


Figure 6.3: Development of the R^2 score of all models on the test set when the feature set is reduced. The black line represents the median, and the gray area represents the 95% confidence interval.

(19 in Diffusion, 32 in Lithography), leaving the models with 20 input features on average, compared to 42 features initially (52% reduction).

As depicted in Figure 6.4, the feature set used in different production areas of the fab can differ. Nevertheless, it can be observed that all features are relevant for a significant fraction of operations in our data. Even the least used feature w is used in over 34% (34% in Diffusion, 36% in Lithography) of the models. This reflects not only the complexity of the HMLV fab, but also the heterogenity of inputs needed to make accurate predictions. However, some noticeable differences in feature importance can be observed when comparing the two production areas in our study. It is remarkable that three of the four most used features are $wt_{dist|t_0}$ (89% in Diffusion, 86% in Lithography), WIP_{dist} (81% in Diffusion, 86% in

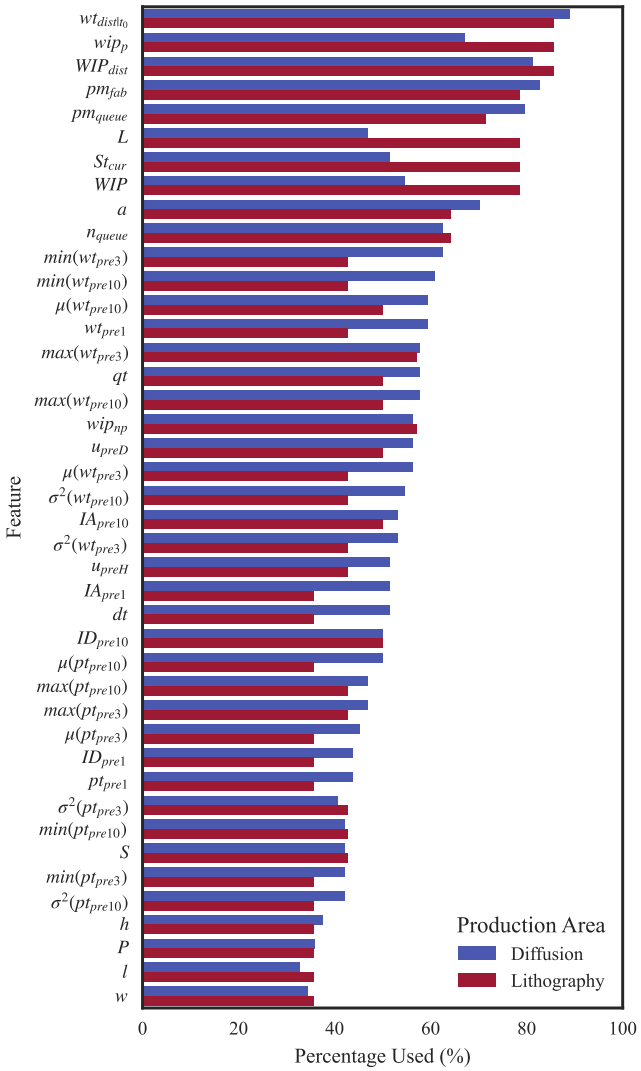


Figure 6.4: Feature usage in percent by production area of the operation. All new features are marked with bold.

Lithography) and pm_{queue} (80% in Diffusion, 71% in Lithography), which are newly introduced features.

Finally, it is remarkable that P (39% in Diffusion, 36% in Lithography) has a relatively low feature importance, compared with their importance both in past publications and production planning processes. While this feature is important for the overall fab performance, our results suggest that it lacks information relevant for predicting single operation waiting time.

We also studied the difference in feature usage by production area. We consider difference in feature usage significant if the usage differs by more than 10% between production areas. The following explanations for those differences were developed in collaboration with line experts. The complete list of those significant differences is provided in Table 6.5. As can be observed, those features can be distinguished by the production area where the feature is more important.

The features more relevant in Diffusion are time-related features, providing inputs on the waiting ($min(wt_{pre3}$, $min(wt_{pre10}$, wt_{pre1}, $\mu(wt_{pre3})$, $\sigma^2(wt_{pre10})$ and $\sigma^2(wt_{pre3})$), the processing ($\mu(pt_{pre10})$) and the arrival (IA_{pre1}) and departure times (dt). Their relative importance stems from higher range of processing times in Diffusion due to batch building, which increases their impact.

Lithography operations react more directly to WIP-related features such as WIP and wip_p. Lithography is the central work center in the fab, which comes along with a higher connectivity to all other fab production areas. Hence, models in Lithography will be more sensitive to a feature providing details about the overall product mix in the fab. Additionally, features indicating the position of the lot in the process, such as L and St_{cur}, seem to be of greater interest in this production area. Since more advanced planning tools are used within Lithography, the system intends to optimize the throughput. Hence, the position of a lot within its own completion process is used by the dispatching system, which results in a greater importance of those features within Lithography. In summary, reasonable qualitative explanations for most observed differences in feature importance between the two production areas could be found.

Table 6.5: Difference in feature usage by production area

Production Area	Feature	Difference
Diffusion	$min(wt_{pre3})$	20%
	$min(wt_{pre10})$	18%
	wt_{pre1}	17%
	IA_{pre1}	16%
	dt	16%
	$\mu(pt_{pre10})$	14%
	$\mu(wt_{pre3})$	14%
	$\sigma^2(wt_{pre10})$	12%
	$\sigma^2(wt_{pre3})$	10%
Lithography	L	32%
	St_{cur}	27%
	WIP	24%
	wip_p	19%

6.4 Discussion

The purpose of this study is to analyze possible features influencing the waiting times in a HMLV fab. To the best of our knowledge, this is the first analysis of this kind yet. The originality of our study lies mainly in the following three points: First, real operational data from a HMLV wafer fab has been used to examine the feature importance. Second, a large selection of features gathered from a thorough review of previous studies has been applied and expanded with features derived from operational experience. Third, feature selection methods were investigated and a framework is proposed that allows to train performant models for different products and operations without the need of manual feature selection. By studying feature importance down to the level of product-operation-combination, we are

able to show that feature importance differs between operations and work centers. [1] It could be shown that a variety of features is needed to make accurate waiting time predictions in a HMLV fab, including features which were first introduced in this study. We hope to provide the scientific community with features that will also prove useful in other studies.

While this study was conducted to the best of our knowledge, there are some limitations to it, which need to be discussed. First, it has to be mentioned that if we make a statement that a particular feature is not commonly used in our models, it does not necessarily mean that this feature does not contain any useful information for such a prediction. The applied feature reduction method filters – apart from features that do not contain any information relevant for the prediction – features that contain redundant information. In doing so, the feature information of for example the lot priority could be included in other features already. Second, the data set used in this study is from one particular wafer fab. Hence, the influencing features could differ considering other fabs, especially when they diverge in terms of automation, size and dispatching policies. Therefore, it would be of great interest if the proposed methodology was used to evaluate feature importance in a low mix fab and to compare those results. Another limitation of this study is the generally low prediction accuracy of the developed models. As described in Section 6.2.2.2, the models were eligible for further analysis if their R^2 score is higher than 0.3. One could argue that this is too low, leaving room for interpretation whether the resulting feature importance is valid, but the authors feel that the innate variability of the data does not allow for more accurate predictions. It would be of interest to analyze the confidence intervals of thus produced predictions, which could be a possible follow-up study.

Another limitation is due to the fact that we had to stick with the data available in this specific use case environment. Hence, there are several features which

[1] It is possible to predict total waiting times with one, monolithic model, as has been shown in the literature. However, this would require much more complex methods to evaluate feature importance between different products, operations and work centers, which is the main goal of this publication.

are quite obviously important, but cannot be taken into consideration. The two main reasons for ommitting features are the sensitivity of personal information like shift assignments and qualification and missing links in real world database systems. Hence, information from the AMHS, as for example used in Wang et al. (2020) are not present in the data.

While the methodology was designed to our best knowledge and delivers valuable results, one can criticize their conception. This criticism can be distinguished into three part. First, there are other options how to design the feature selection method. Applying other feature selection methods and comparing them with our approach is conceivable. Recently, Wang et al. (2020) reported that applying a network based key factor identification approach to obtain the direct correlation between influencing factors resulted in a higher predicting accuracy on cycle time than a prediction model with all factors. It would be very interesting to obtain the performance of this approach in our HMLV fab and compare it to the results achieved in this publication. Second, the chosen focus on the mentioned modeling technique can be criticized, since other statistical approaches such as Kernel Ridge Regression, Bayesian Regression or Stochastic Gradient Descent may outperform our models in terms of forecasting accuracy. It would be of great interest if the appropriateness of our choice would be challenged in future publications. Third, the training methods used in our methodology can be designed differently. We tried another method in the setup phase of this study to observe whether random and time-related splits perform differently. Nonetheless, it would be of interest to conduct experiments about the impact of the train-test-validation split ratios. To keep the focus of this study, we fixed those variables to common practice values, but other configurations may lead to better results. Hence, again, this would be of great interest to investigate in future publications.

Finally, as already mentioned in Section 6.3, the explanations provided for the deviations in feature importance between the production areas are only qualitative and lack measurability. It would be of great interest to enhance this methodology to enable a standardized root cause analysis of the results. However, this is not possible yet to our knowledge.

6.5 Managerial Implications

The results of our study indicate not only the heterogenity of information necessities, but also the increasing needs for improved data quality and advanced prediction features when controlling HMLV fabs. Scheduling decisions and cycle time predictions which are based on the same feature sets as in low-mix fabs will inevitably result in non-optimal fab utilization and consequently in cost inefficiencies and delayed supply. To our knowledge, the data required to compute the novel features is already available. However, as the features themselves are computed in normal operation, their subsequent calculation is complex and time-consuming. Therefore, we encourage decision makers to invest in and maintain improved and extensive data warehousing to facilitate employing the beforementioned and other possible features for predictions.

6.6 Conclusion

The objective of this study is to provide a holistic overview of influencing features on the waiting times in a HMLV semiconductor wafer fabrication and to analyze those features regarding their importance. Three conclusions can be drawn from this study. First, we were able to prove the importance of all features in the context of our HMLV fab. We demonstrate that each feature is relevant for waiting time prediction which highlights the extensive demand for information when controlling HMLV fabs. At the same time, however, we show that feature set reduction is beneficial for the precision of our prediction models. Second, our results indicate that also the novel features introduced in this study have significant impact on the waiting time predictions in the semiconductor wafer fabrication and should therefore be included in future methodologies to predict cycle- and waiting times. Third, we have shown that it is possible to include all features used in the proposed methodology and find the optimal set of features for each product-operation combination automatically. Nonetheless, regardless of the number of features used or the model complexity, a prediction gap remains. The complexity

of the semiconductor wafer fabrication with thousands of mechanical, human and process-related influencing features can not be covered completely by a model using the currently available data. Therefore, it will be of interest to see future developments in this field, when the advancing application of industrial internet of things (IIOT) will further increase data availability.

In future research it would be of great interest to challenge the limitations discussed above. Therefore, the performance of the proposed system with other modeling techniques and larger or different data sources is a valid next step for further research. The ultimate goal is predicting lot cycle times. While the waiting times are of interest for internal planning, the information relevant for customers is the influence on the completion date. Hence, we will focus our next studies on the connection of the models developed in this study to come up with a framework for lot cycle time predictions.

7 Route estimation

In the previous Chapter 6, we have developed and tuned models to estimate waiting times of operations. Since the ultimate goal is to estimate waiting times – and approximate processing times – with models for each operation along a route, so that in the end those estimations can be summed for a cycle time estimation, the next question is to identify the routes for a lot. As explained in Section 4.1, the estimation of the route a lot will take is a non-trivial task. Unlike waiting times, which – as shown in Chapter 6 – can be estimated using a set of influencing features, route estimations depend on features which are not known yet. Thus, the approach chosen in this thesis is to draw a route based on the route probabilities. While it is a potential connection point for future research to develop a machine learning based route approximation, this is not in the scope of this thesis. In order to draw a route based on the route probabilities, we introduce a methodology to identify the set of routes and the route probabilities.

As mentioned before, routes consist of a sequence of operations within a stage. The goal is to identify all possible routes and quantify their probabilities. A route probability is computed by calculating the fraction of lots of a product type that have taken the route divided by all lots of a product type for a given time period, using the lot traces in this time period. Thus, the probabilities of all routes sum up to 1 for each stage. Two things have to taken into consideration for this calculation:

1. The eligible sequences of operations are constantly changing over time, mostly due to the work of quality and product engineers that are working on process and quality improvements. Their updating time is tracked in a database. Therefore, when identifying routes and their probabilities, it is not valid to use the complete history of operation sequences in a stage, as

most of them are not possible anymore. Those routes that are possible at a point in time are called **active routes**.

2. When deriving the route data the active routes are available for this time point. Nonetheless, there is no history of active routes available in the given use case. Consequently, it is not possible to know the active routes for each time point, just for the snapshot when the data was acquired. Nonetheless, the history of routes of past lots – which we call **historic routes** – is known.

To cope with these problems, we create a two-fold approach: One algorithm makes use of the currently active routes and is therefore intended to be used in operations. This approach is introduced in the upcoming Section 7.1. Additionally, another algorithm approximates the active routes for any given time point by using the historic routes and is therefore used for the validation of the cycle time estimation methodology, where cycle times from lots in the past are estimated and compared. This approach is introduced in Section 7.2.

7.1 The active route approximator

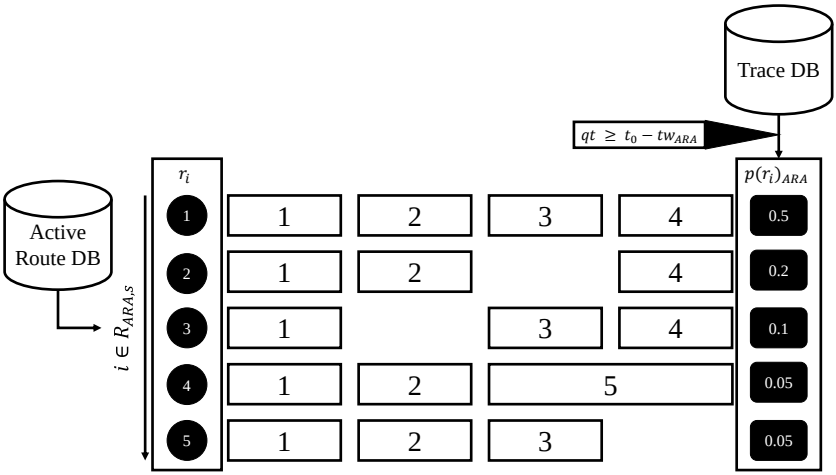


Figure 7.1: ARA: Using active routes $R_{ARA,s}$, while the probabilities $p(r_{active})$ are calculated based on the traces, filtering the lots within the traces by a time window tw_{ARA}. This time window is determined by the last route update of any route in the stage. The lot start qt has then to be within this time window.

For operational use, we develop an algorithm to identify the currently active routes. In the following, this algorithm is called "active route approximator" (ARA). Its basic functionality of choosing the possible routes from a different source than their probabilities is shown in Figure 7.1. It looks up all active routes and then calculates the route probabilities within a given time window tw_{ARA}. This time window is determined by the aforementioned newest updating time of any route in a stage. This time point t_s is defined as follows, with $t_s(r)$ as the point in time when the route r_i out of R possible routes in stage s got its latest update:

$$t_s = max(t_s(r_1), t_s(r_2), ..., t_s(r_R)) \tag{7.1}$$

The issue with t_s is that all routes are equally important. Consequently, it could happen that a route which occurs rarely could remove a majority of the lots from the data set. Therefore, a threshold is introduced, on which routes are considered to define t_s. We evaluate three policies how to choose the route set for each stage:

- **Top 5:** For each stage, the five routes with the highest probability in the considered time window are chosen. This sets an upper bound to the calculation complexity for every stage, but could fail to cover a large amount of routes that were taken by a significant fraction of lots in the considered time window.
- **0.1% probability:** For each stage, all routes are chosen which were used for more than 0.1% of all lots in the data from the considered time window. This policy removes those routes which are very unlikely to be followed. Nevertheless, it is still possible that the route set has a high extent (leading to higher calculation times) or the coverage is not sufficient, if the tail of route probabilities is long.
- **>99% coverage:** For each stage, we add the routes in descending order of their rate of occurrence to the route set, until a coverage of 99% is reached. This policy installs a lower bound to the coverage, but can result in big route sets, which ultimately leads to long calculation times.

The route reduction policy is implemented in the methodology as a parameter, but since the ARA is of limited use for the validation of the cycle time estimation methodology, we are not covering a numerical analysis of the route selection policy within this thesis. Nevertheless, the coverage of routes using the different policies is shown in Figure A.1, showing that even with the Top 5 approach, over 90% of all lots are covered for our dataset. Thus, we propose to apply this policy during operational use, since if leaves the most lots in the data set and therefore increases the accuracy of the route probabilities at the expense of ignoring infrequent routes.

7.2 The historic route approximator

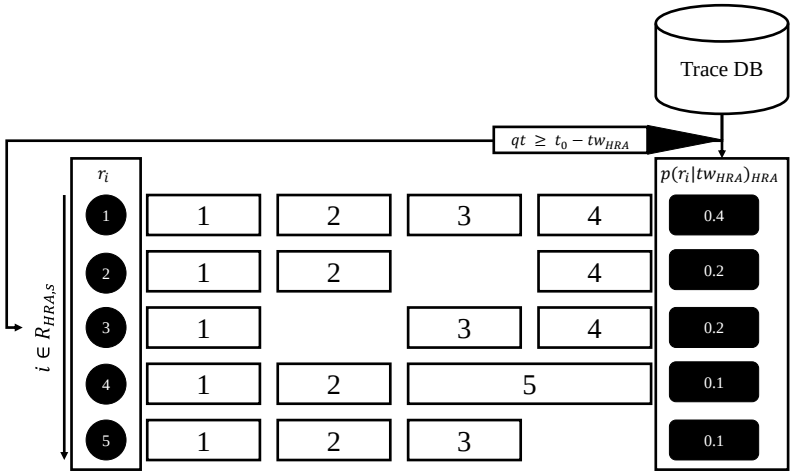


Figure 7.2: HRA: Using the possible route set $R_{HRA,s}$ as well as their probabilities $p(r_{historic})$ based on the traces, which are filtered based on a time window tw_{HRA}. tw_{HRA} is a free parameter and has to be set.

Since the active routes are not known for any given time point, a methodology is needed to approximate them. For the purpose of the analysis of the cycle time estimation of the proposed methodology, which is done based on past lot cycle times – and therefore also past and possibly outdated routes – we establish another algorithm, which approximates the possible routes and their probabilities at any given time. This algorithm is called "historic route approximator" (HRA). Its basic functionality is visualized in Figure 7.2. It approximates all active routes and their probabilities purely based on their occurrence in a time window tw_{HRA} right before the lot start, which enables route estimations at any point in time. Nonetheless, it has to be highlighted that this approach is only an approximation for the actually available routes at any given time. It is possible that major route changes occur within this time window and therefore routes are chosen that are

not available anymore. For HRA, we can not use route updating times to estimate the time window, because they are not known for non-active routes. Therefore, the goal is to find an optimal time window tw_{HRA}, for which we propose a methodology in the following.

To determine the time window tw_{HRA}, we compare the route probabilities of ARA $p(r_{ARA})$ for the time point where the data set was extracted with the route probabilities for the same time point using the HRA – with time window tw_{HRA} – $p(r_{HRA}|tw_{HRA})$. Thus, we check the deviation of HRA from ARA and try to minimize this deviation by choosing a time window. This is the only point in time where the actually active routes are known. This results in the difference $\Delta(p|tw_{HRA})_s$. Hence, we use the probabilities from the ARA from this point in time to compare them with the probabilities of the HRA, using a time window tw_{HRA}. This process is shown in Figure 7.3.

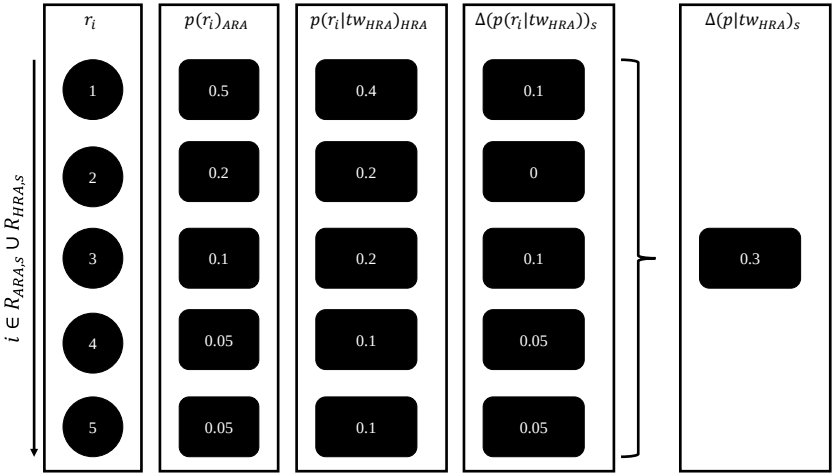


Figure 7.3: Determining the probability difference $\Delta(p|tw_{HRA})_s$ of ARA and HRA for a given time window tw_{HRA} in a stage s by calculating the mean deviations of the probabilities of ARA ($p(r_{ARA})$) and the probabilities of HRA ($p(r_{HRA}|tw_{HRA})$) for every route $r_i \in R_{ARA,s} \cup R_{HRA,s}$.

Since the route sets from both the ARA and the HRA are combined, probabilities of 0 for some routes can occur for both methodologies. We now choose tw_{HRA} so that the difference $\Delta(p|tw_{HRA})_s$ gets minimal. This is done by applying different values for the tw_{HRA} and calculating the corresponding difference.

It has to be underlined that the definition of the optimal time window for the HRA is a free parameter, which can affect the performance of the method. Since the active routes are only available for the time point when the data was acquired – and therefore no history of active routes is available – there is only one time point where those two approaches can be compared. The optimality of the resulting time window is therefore not provable given our available data and cannot be generalized. It is possible to collect the active routes over a longer period of time and therefore create the foundation for a more detailed analysis of the optimal time window for the HRA, but this is not in the scope of this thesis. This can be a connection point for future research, since it would enable the proof of the generalization capability of the HRA.

While the HRA is a necessity to evaluate the given case, it is recommended to use ARA when deploying the proposed methodology. The approximation of the optimal time window for HRA in the given case is executed in Chapter 9.

We have now introduced all aspects of the methodology. Before the methodology can be applied to real operational data, the issue has to be resolved that feature values are not known at any point in time, which was assumed in Chapter 6. Therefore, we introduce techniques to deal with feature uncertainty in the following.

8 Feature estimation

In the two previous chapters, methods to estimate waiting times and to estimate the route of a lot were presented. In this chapter, we will introduce ways to estimate feature values under uncertainty, which is necessary for waiting time estimations along a route. In Chapter 6 a broad variety of features has been introduced and is further considered for estimation. The impact of those features on the estimation capabilities of models has been evaluated under the assumption that the value of each feature is known for every point in time. While this is true for the past – which made it possible to train models on it – this assumption does not hold for estimations in the future. If a lot is started at $t = 0$, features are measurable for this time point. Afterwards, they can only be estimated. Different strategies for feature estimation are needed, depending on the data source for the features and their types. We present in Section 8.1 how the features are are grouped by type and what needs to be considered for each of those groups. Afterwards, we propose several feature estimation methods for the future in Section 8.2.

8.1 Feature types

Features are distinguished into four groups: route-based features, lot history-dependent features, equipment-group history dependent features and fab history-dependent features. Those feature groups are presented in the following. The features which belong in each group are named and **highlighted**, while their definitions were presented in Chapter 6. Unless noted otherwise, the definition and computation of those features has not been changed compared with Chapter 6.

The features, their categorization and the corresponding estimation method are summarized in Table 8.1.

Table 8.1: Features, their categorization in Deterministic (1), Lot history-dependent (2), Equipment group history-dependent (3) and fab history-dependent, as well as the corresponding estimation method.

Feature	Categorization	Estimation method
Priority	*1*	
WIP in fab	4	Section 8.2
WIP in queue (prod./non-prod.)	3	Section 8.2
Job Arrival Time	2	Lookup
Inter-Arrival-Time	3	Section 8.2
Inter-Departure-Time	3	Section 8.2
Machine (group) utilization	3	Section 8.2
Machine (group) availability	3	Section 8.2
Prev. (1/3/10) waiting time	3	Section 8.2
Prev. (1/3/10) processing time	3	Section 8.2
Nbr. of operation loops	1	
Product mix in fab	4	Section 8.2
Product mix in queue	3	Section 8.2
Nbr. of different products in the queue	3	Section 8.2
Time since last departure	3	Section 8.2
WIP profile	4	Section 8.2
Similar operations waiting	3	Section 8.2
Waiting times at t_0 in queue	3	Section 8.2
Level of completion of lot at t_0	1	
Shift	2	Lookup
Weekend	2	Lookup
Holidays	2	Lookup
Previous operation	1	
Layer	1	
Stage	1	

8.1.1 Deterministic features

A feature is perceived as deterministic, if it is known a priori or once the route is drawn. As described in Chapter 7, each lot will take a route option. While the route is not known beforehand, some features are known at the time point when the route is drawn, because it is determined by the route. The **previous operation** of an operation is determined by the route, as well as the **level of completion** of a lot at any given time point. Furthermore, the **number of operation loops** is known once the route is drawn. Additionally, we use features whose values are deterministic anyway. The **Stage** and the **Layer** are always known, irrelevant of the chosen route. Features belonging in this group do not need further uncertainty handling, because they can be assumed to be known. Thus, we treat those features accordingly.

8.1.1.1 A special case: The lot priority

The lot **priority** is a special case of an impact factor. This feature is the main leverage point for line controllers in SC manufacturing environments. In theory, the priority gets assigned a priori before lot execution starts, which would make the priority a route-based feature, as described in Section 8.1.1. During lot execution, the priority can be changed. The most common reason is an upcoming demand due date, which consequently triggers a priority increase by a line controller. This is numerically analyzed and displayed in the Appendix (Figure A.2), using the data to be used in Chapter 9. While there are changes in the priority during the lot execution, it is not in the scope of this thesis to include them in the model. Consequently, the lot priority is handled as a route-based feature, assuming that the priority is not adjusted during lot execution. Hence, we do not try to estimate interventions by production control within this thesis. Hence, it is used as a connecting point for future research.

8.1.2 Lot history-dependent features

The second group of features depend on the lot history. Unlike route-based features, lot history-dependent features are not known once the route is drawn, but are assumed to be exactly determined when all calculations before this point were correct. For example, the time point when a lot enters a certain queue and the **job arrival time** are deterministic, if all previous waiting and processing time estimations are correct. If so, the **shift** in which an operation is executed is also known. Similarly, whether the operation enters the queue on a **weekend** and/or on **national holidays** is known.

We decide to look these features up at the point in time when all previous predictions of this scenario indicate it to be, because it seems to be the most valid guesser. A further randomization of these features is not expected to be beneficial. Hence, the optimal strategy to cope with uncertainty for those features is to look them up at the point in time where the lot history indicates them to be executed.

8.1.3 Equipment-group history-dependent features

The third group of features depend on the equipment-group history. Unlike the first two feature groups, features which are equipment-group history dependent are not dependent on the lot for which the estimation is made, but on the equipments capable of executing an operation at a certain time point. The features are calculated for every full hour of the data set. Thus, arrival times of lots are binned for the lookup. These features reflect the past behavior of these equipments. It includes all **previous waiting time** and **previous processing time** features, as well as all features related to **inter-arrival time** and **inter-departure time** (including time since last departure), as well as all features describing the queue (**product mix in the queue**, **waiting time distribution in the queue** and **WIP in the queue**) and the **utilization** of equipments. Since these features are highly variable over time, an estimation within a simulation is fairly complicated and detracts from one advantage of the proposed methodology: its calculation speed. Consequently, it

is tested within this thesis whether those features can be approximated by (rolling) averages over the past, or by random draws from their historical distribution. These distribution is flat in time, since the feature values are calculated per full hour. Another question to investigate is whether those feature values shall be drawn independently from each other, or all together, in order to avoid impossible combinations of feature values.

8.1.4 Fab history-dependent features

The fourth group of features depend on the fab history, describing the overall fab status. Features that belong in this group are the **WIP in the fab**, the **product mix in the fab**, as well as the **WIP profile**. They are accumulations of the status of each equipment-group and are measured for set time intervals – in our case full hours.

Similarly to equipment-group history-dependent features, those features are approximated with different uncertainty handling techniques in the course of this study. For operational usage, we assume that it is possible to receive the expected values of such features from other entities, since they are solely dependent on production start planning. The exact movement of lots through the fab is irrelevant for the estimation of those features, so that they should be known in a short-term time window of weeks and at least possible to approximate for mid-term time windows of months.

8.2 Feature estimation strategies

Since it is not possible to know all features at all time points, strategies have to be developed to estimate them at any given time point. As described before, the features can be distinguished by their dependencies, resulting in a need for such a strategy for equipment group history-dependent and fab history-dependent features. We discuss four possible strategies. We will apply those strategies

afterwards in Chapter 9 to evaluate their impact on the predictive capability. From this evaluation we derive a proposal for the preferred strategy.

All strategies have in common that they intend to approximate feature values at any time point by past (and therefore known) values, using either the median, or a random draw within a chosen time window. Additionally, the features are either approximated from the lot start time point (fixed time window), or from each operation start time point (rolling time window). Thus, the strategies can be split up along these parameters, as shown in Table 8.2.

Table 8.2: Overview of the uncertainty handling strategies

		picking strategy	
		median	draw
time window	fixed	Section 8.2.1	Section 8.2.3
	rolling	Section 8.2.2	Section 8.2.4

We discuss these approaches in more detail in the following.

8.2.1 Median feature values in fixed time windows at lot start

Since all feature values after the queue entering time qt are assumed to be unknown, the first strategy is to fix those values to the level they had before this time point. Within this time window, an aggregation method can be chosen for each feature to determine an overall feature value in a given time window. Thus, the longer the time window, the heavier the smoothing. This strategy is shown in Figure 8.1.

The most intuitive aggregation method is the sample mean over the time window $\overline{x}$. As described before, most feature distributions have a tail towards high values, which induces a bias towards higher values when using the mean as the estimator for a given time window. Due to the absence of a dedicated outlier removal

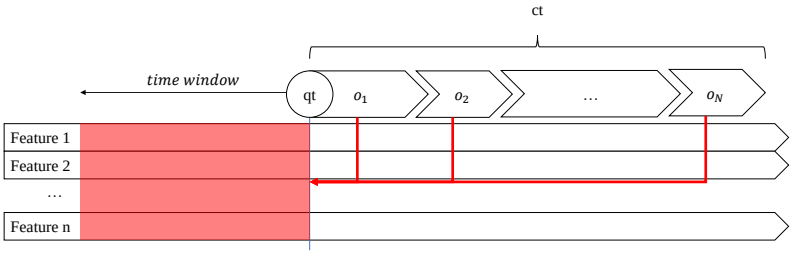


Figure 8.1: Feature approximation approach, which uses an aggregation method in a time window before the lot start qt to estimate the feature values for any operation $o_1, ..., o_N$ along a route.

methodology for features, the median $\widetilde{x}$ instead of the mean is used here as the estimator within a given time window.

Before comparing this method with other methods, the time window has to be defined. Since it is unclear how the choice of time interval impacts the estimation accuracy, several time windows are compared by MAE of the final cycle time prediction to choose the time window with the lowest resulting value. The details of how we arrive at this choice are given in Section 9.6.1.

8.2.2 Median feature values in rolling time window

Another feature estimation method is the usage of rolling time windows when calculating aggregation values of features for all predictions, as it is shown in Figure 8.2. It is expected that this methodology provides more flexibility and more accurate approximations of the feature values for data sets that span a long time period. The calculation of an aggregation value follows the same procedure as for fixed time windows. In the following, the median is used. Similar to the previous approach, the time window has to be investigated. A study of this is presented in Section 9.6.2.

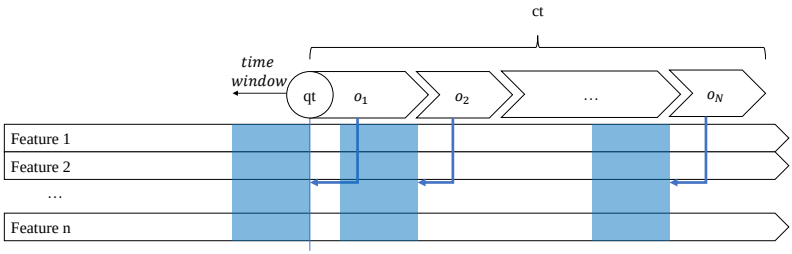


Figure 8.2: Feature approximation approach, which uses an aggregation method in a time window before the operation start to estimate the feature values for any operation $o_1, ..., o_N$ along a route.

It has to be mentioned that a cycle time estimation at lot start is not possible with this approach, because it requires inputs that are not known at this time point. Nonetheless, this approach can be used when the predictions are constantly refreshed, which is of interest for example in dispatching or operation completion estimation. Additionally, this approach provides insights on the solution quality, if the cycle time estimation was connected to a simulation that predicts the feature values. To follow this approach, only a rough estimation of those feature values is needed since this approach uses aggregated instead of exact feature values.

8.2.3 Draw of feature values in fixed time window

Feature values can be drawn from their own historical distributions in a given time window, which is shown in Figure 8.3. Each feature f is represented by a distribution function obtained from histogramming historical feature values per lot. This results in a distribution discrete values distribution function $p(x) = P(x \in S)$, where $S = \{x_1, x_2, ..., x_k\}$ with x_k as possible feature values. S therefore represents all possible feature values for f. If these values are sorted

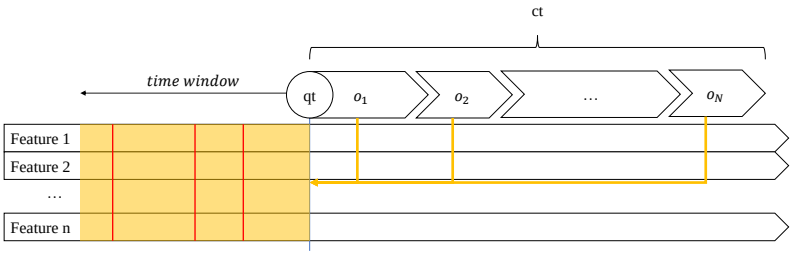


Figure 8.3: Feature approximation approach, which uses a random draw in a time window before the lot start qt to estimate the feature values for any operation $o_1, ..., o_N$ along a route.

in ascending order, the cumulative distribution function (CDF) $F(x_j)$ of feature value x_j is defined as follows:

$$F(x_j) = \sum_{i \leq j} p(x_i) \tag{8.1}$$

By slicing the interval $(0, 1)$ into sub-intervals (partitions) the draw is done based on the CDF-value of the variables:

$$(0, F(x_1), ..., F(x_{k-1}), 1) \tag{8.2}$$

Subsequently, a random variable $U = Uniform(0, 1)$ is drawn, and it is observed, in which partition it is located. Similar to the approaches described in Section 8.2.1 and Section 8.2.2, the time window of the value draw is up to further evaluation. A detailed numerical analysis on finding the optimal time window for this purpose is conducted in Section 9.6.3.

Another option is to draw a lot from the past and take the feature values from it, instead of drawing the values independently from each other. This avoids impossible combinations of values while limiting the randomness of the approach. This approach is also evaluated in Section 9.6.3.

8.2.4 Draw of feature values in rolling time window

Ultimately, it is also possible to use the rolling approach in combination with a draw of the feature values, as shown in Figure 8.4. Its mathematical foundation is the same as explained in Section 8.2.3. Within this approach, a value is drawn

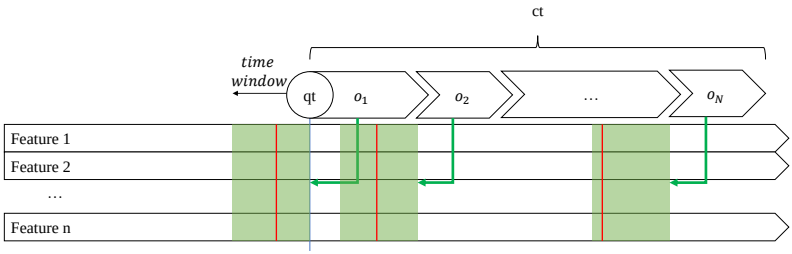


Figure 8.4: Feature approximation approach, which uses a random draw in a time window before each operation start to estimate the feature values for any operation $o_1, \ldots, o_N$ along a route.

not in the fixed time window before qt, but in a time window before the queue time of every operation. Again, the optimal time window has to be found, which is performed in Section 9.6.4.

Also, as for the approach described in Section 8.2.3, it is an option to draw the feature values based on a lot from the past instead of independently choosing them, as this possibly avoids impossible combinations of feature values. This approach is also evaluated in Section 9.6.4.

9 Numerical evaluation

We have elaborated a methodology to train and optimize models for each operation, a methodology to identify the route the lot will take, and several methodologies to estimate feature values. If waiting times, in combination with approximated processing times are accumulated along a route, entire cycle times can be estimated, which results in the methodology described in Section 6.2.

In the following, in Section 9.1, we provide a set of experiments to evaluate the impact of multiple facets proposed methodology:

- The waiting time estimation, as presented in Chapter 6.
- The route estimation, as presented in Chapter 7.
- The feature estimation, as presented in Chapter 8. This again splits up in the different estimation methods presented in this Chapter.

Subsequently, in Section 9.2, we present the data set, which is used for the analysis. Then, in Section 9.3, we investigate the choice of a time window for the route approximation. Afterwards, we present and analyze the experiments introduced in Section 9.1. This includes both a detailed analysis of each experiment (Section 9.4-Section 9.6), as well as a comparative analysis of results, leading to an optimal configuration of the methodology depending on the considered use case. Even though the goal of this methodology is to enable the estimation of entire cycle times, there are still other use cases, such as mid- or short-term planning or just-in-time predictions, which could make use of other configurations than the ones using both the feature and the route estimation. The chapter is

concluded by a comparative analysis of the methodology (in its chosen configuration) with industrial standard approaches in Section 9.7, which were introduced in Section 4.2.

9.1 Design of experiments

Table 9.1: Set of experiments conducted within this thesis.

Experiment	Route	Features	Time window	Section
0	known	lookup	–	Section 9.4
1	draw	lookup	–	Section 9.5
2	draw	median	fixed	Section 9.6.1
3	draw	median	rolling	Section 9.6.2
4	draw	draw	fixed	Section 9.6.3
5	draw	draw	rolling	Section 9.6.4

Within this section, we conduct a set of experiments to evaluate the performance of the developed methodology for total cycle time estimation and to analyze potential future areas for improvement. All experiments are analyzed based on the KPIs introduced in Section 2.4. We conduct these KPIs not only for all lots within the validation data set, but also for all lots within this set, which result in a cycle time of less than 35 days (low cycle time), as well as all lots within this set with a resulting cycle time of more than 48 days (high cycle time). With that, we intend to analyze the estimation capability of the methodology for lots deviating from mean cycle times. We consider two degrees of freedom in this methodology: First, whether the route is known. If not, the route approximation algorithm developed in Chapter 7 is applied. Second, whether the feature set is known. If not, the feature approximation algorithms developed in Chapter 8 are applied.

We set up a baseline model, where both degrees of freedom are fixed and the start time of each stage is known. In doing so, an estimation based solely on historical data is made. For simplicity, this baseline model is called **Experiment 0** from now on. Afterwards, we analyze the impact of the route approximation by including the route approximation algorithm (**Experiment 1**) and analyzing the effect of route uncertainty on the estimation quality compared to the baseline model. In the remaining set of experiments, we evaluate the impact of the feature approximation algorithm, using median feature values in a fixed time window (**Experiment 2**), median feature values in a rolling time window (**Experiment 3**), draw of feature values in a fixed time window (**Experiment 4**) and a draw of feature values in a rolling time window (**Experiment 5**). For each experiment we conduct the aforementioned **two-sided t-test** to evaluate whether the estimation is significantly biased. Thus, the null hypothesis for every Experiment x is as follows:

$\mathbf{H_0}$: The distribution mean of Experiment x equals the distribution mean of the actual cycle time.

We reject the null hypothesis with a significance $\alpha = 0.05$, representing that, if the null hypothesis is correct, random draws from both distributions share the same distribution mean.

Afterwards, we compare these approaches with each other to identify the optimal strategy and with Experiment 0 and 1 to evaluate the impact of the feature approximation. Within this step, we apply a **one-sided t-test** to the residual distribution of each experiment x and the residual distribution of each other experiment y to investigate whether the mean absolute estimation error of one experiment is significantly higher than from another experiment. Therefore, the null hypothesis is formulated as follows:

$\mathbf{H_0}$: The mean absolute error distribution Experiment x is greater or equal than the mean absolute error distribution of Experiment y.

We reject the null hypothesis with a significance $\alpha = 0.05$, representing that, if the null hypothesis is correct, random draws from the residuals of Experiment x result in a significantly higher MAE than from Experiment y.

Ultimately, we validate it based on the industrial standard approaches introduced in Section 4.2. We give an overview of all experiments in Table 9.1. Also here, we use the **one-sided t-test** to validate our findings.

We provide the analysis of each experiment in the following sections. While we indicate the aforementioned KPIs and hypothesis tests that are relevant for each section within the section, we give a complete tabular overview of all KPIs and t-test results for all conducted experiments in Appendix D.

9.2 Data Set

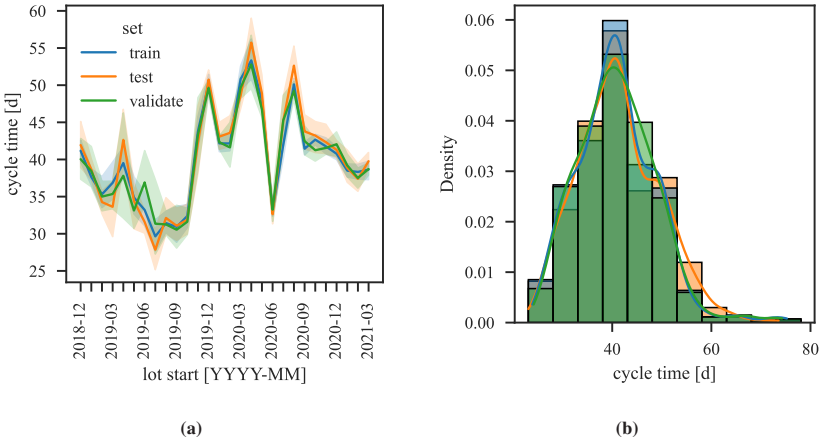


Figure 9.1: (a): Average cycle time per month of the chosen product over time and the interval of the 5-95% percentiles separated by the data subsets. (b): Density histogram of cycle times for the chosen product separated by the data subsets.

For the numerical evaluation, we use operational data from a SC wafer fab of the Robert Bosch GmbH, as in Chapter 6, using also the same input features. Thus, the input feature definition and calculation has not changed. The only difference is that the following analysis is performed only on one high-runner product, whereas the analysis in Chapter 6 was done on 30 high-runner products. The data is comprised of the feature values, waiting and processing times for all operations as well as timestamps when these operations occurred. These data are collected for all lots starting between December 2018 until March 2021. All lots executed and finished within this time frame are randomly split into three groups: First, 60 % of the lots are taken to build the training set, with which all models are trained, as explained in Section 6.2.2.1. Second, 20 % of the lots are used to build the test set, with which hyper-parameter tuning and the feature set reduction explained in Section 6.2.2.3 are executed. The remaining 20 % of the lots are used for the numerical evaluation. By doing so, we avoid including lots in the performance analysis that the models have already seen in training.

The average cycle time of the chosen product is shown in Figure 9.1 (a). It becomes clear that – despite a constant heavy variation – the average cycle time is prone to underlying trends during some time spans within the data set, for example a decreasing period during the first nine months. Thus, we use a random split of lots to avoid that a data subset only covers parts of that trend. Additionally, it can be observed that the average cycle time in all three data sets vary similarly, indicating that the positional parameters of the sets are comparable. The distribution of cycle time within all three data subset is shown in Figure 9.1 (b). No significant difference in the distribution is visible.

9.3 Route time window determination

As mentioned in Chapter 7, the optimal time window for the route approximation algorithm has to be defined. To limit the scope of this analysis, we fix the time window for ARA tw_{ARA} to 120 days, and only optimize the time window for HRA tw_{HRA}.

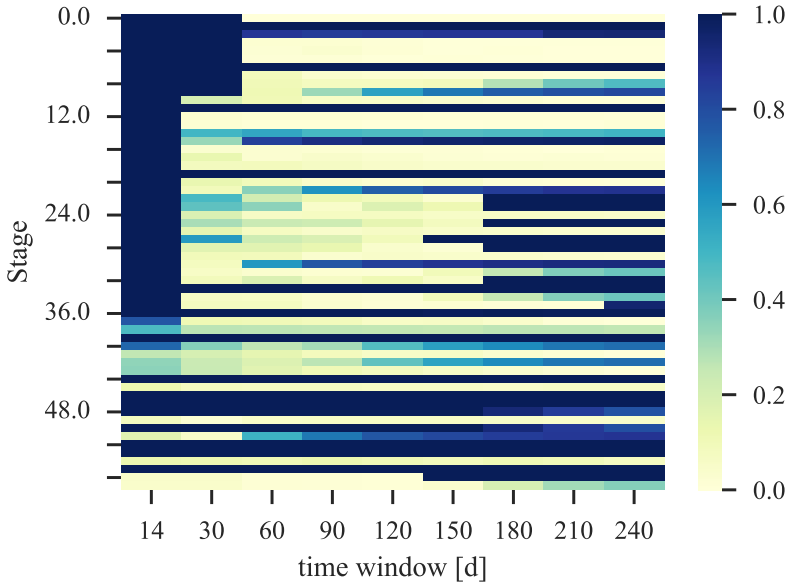


Figure 9.2: Stage-wise deviation of the route probabilities for different time windows

Figure 9.2 visualizes the deviations between the route probabilities of the active routes and the historical routes with several time windows. As visible, there are several stages where deviation is constantly 1, irrelevant of the time window. These stages are the ones where the operation set has recently been changed. Thus, no similarity in the routes from ARA and HRA in these stages are observed. The mean squared errors (MSE), as well as the sum, squared errors (SSE) for each time window are visualized in Table 9.2. The differences in deviation due to different time window choices are negligible (all time windows between 60 and 240 days result in similar MSE-values), and – as already mentioned – the methodology does not result in a global optimum, since it can only use one time point (the point where the data is acquired). Nevertheless, we choose a route time window of 60 days based on the MSE and SSE value comparison.

Table 9.2: Mean squared error (MSE) and Sum of squared errors (SSE) of the different time windows for the route approximation methodology.

tw [d]	14	30	60	90	120	150	180	210	240
MSE	0.75	0.32	0.21	0.22	0.23	0.22	0.22	0.22	0.22
SSE	43.30	18.30	12.29	12.78	13.53	12.68	12.35	12.60	12.61

9.4 Experiment 0: Baseline estimation on historical data

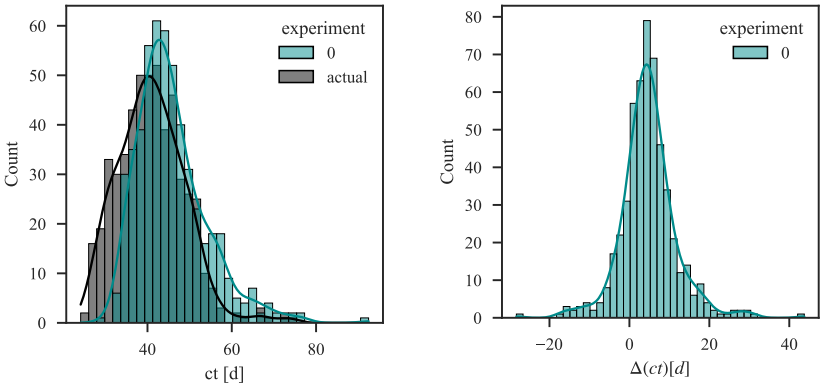
To evaluate the capability of a network of models while all degrees of freedom are perfectly determined, a baseline estimation is introduced. In this baseline prediction, which is called **Experiment 0** in the following, the route as well as the feature set at each time point are known. Additionally, to avoid a systematical shift in the look ups, we also assume that the stage start times are known. Hence, deviations in the predicted sojourn times will not sum up to a mismatch. Apart from providing a baseline for the evaluation of all other experiments, this set up is applicable if the goal is to estimate cycle times historically, or to do just-in-time estimations of individual waiting times. The histogram of the predicted and the actual cycle time of the validation lots is presented in Figure 9.3. It is distinguished in absolute values (figure a) and error values (figure b).

Table 9.3: Key performance metrics for Experiment 0 for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
0	45.75	44.33	0.36	8.33	9.47	0.0	4.76	6.38	8.45

As visible in Table 9.3 an MAE of 6.38 days is observed, meaning an average deviation of approximately a week. The ME of 4.76, representing the bias of the estimation, indicates that a positive systematical error is included in the deviation. This is underlined by the p-value of the two-sided t-test, indicating

that the hypothesis that the means from Experiment 0 and actual are identical can be neglected. This is also visible in the histograms. Thus, a $\delta(MAE)$ of 75% indicates that 75% of the variance can be explained with the systematical error, while 25% is due to unexplained variance.



(a) Absolute **(b)** Relative

Figure 9.3: Distribution of predictions for Experiment 0

The development of the cycle time using Experiment 0 compared to the actual cycle time development is visualized in Figure 9.4. The seasonal and overall trends are similar in both lines, even though the mean of Experiment 0 is nearly all the time higher than the mean of the actual values, indicating the systematical error mentioned before. Nevertheless, it is observable that the error of the estimation is significantly higher in 2019 and 2020 than afterwards. This is also underlined by the study of KPIs for lots in 2021 shown in Table D.7. Thus, it is possible that the observed correlations between the input features and the waiting times in this time span were different than the ones afterwards, leading to the question whether a reduction of the time span could significantly reduce the estimation error. We discuss this question in Section 10.1.

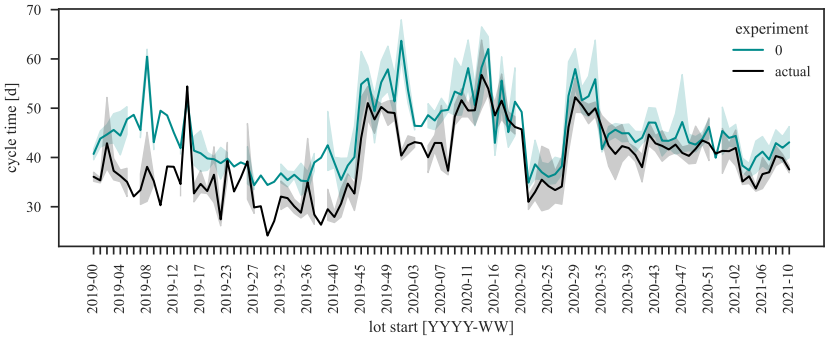
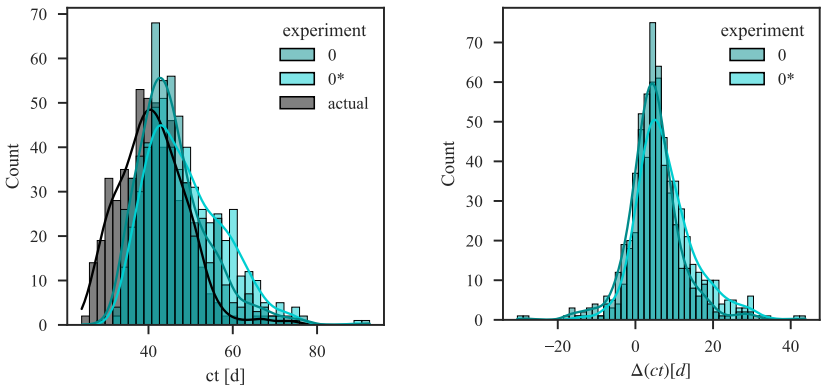


Figure 9.4: Average cycle time over time for Experiment 0 and the actual cycle time.

It is also possible to analyze the behavior when the stage start times are not known. By doing this, we extract the impact of the knowledge of the start times for the estimation quality. We call this sub-experiment **Experiment 0***. We expect that if stage start times are not known the estimation accuracy decreases, because errors over time sum up, leading to more incorrect lookups of feature values for later operations. The corresponding histograms are visualized in Figure 9.5. The cycle time development comparison of those Experiments would exceed the scope of this thesis, but is shown in Appendix C.



(a) Absolute

(b) Relative

Figure 9.5: Distribution of predictions for Experiment 0 and Experiment 0 without the knowledge of the stage start times (Experiment 0*)

As visible in Table 9.4, without the information about the stage start times, the MAE increases to 8.42, and the ME to 7.61, which indicates that the absolute error mainly derives from a systematical error. Accordingly, the null hypothesis of the corresponding two-sided t-test can be rejected.

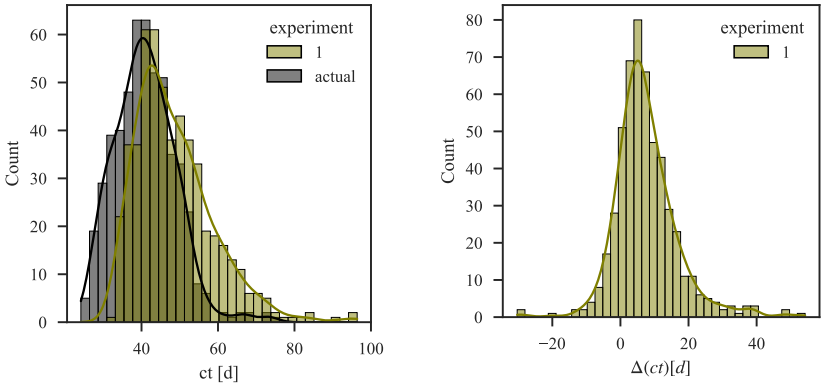
Table 9.4: Key performance metrics for Experiment 0 and 0* for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
0	45.75	44.33	0.36	8.33	9.47	0.0	4.76	6.38	8.45
0*	48.6	47.17	0.39	9.04	14.49	0.0	7.61	8.42	10.86

9.5 Experiment 1: Impact of route uncertainty

The objective of Experiment 1 is the extraction of the impact of the route estimation algorithm developed in Chapter 7. Therefore, the route is unknown, while all

feature values are assumed to be known. By comparing with Experiment 0, the impact of the route parameter and the performance of the route estimation algorithm are analyzed.



(a) Absolute **(b)** Relative

Figure 9.6: Distribution of predictions for Experiment 1

As visible in Figure 9.6, a similar behavior as in Experiment 0 can be observed. Again, the predicted cycle times seem to be slightly higher than the actual cycle times. The development of the estimated cycle times using Experiment 1 compared to actual cycle times is visualized in Figure 9.7. The same conclusions as for Experiment 0 can be drawn.

Table 9.5: Key performance metrics for Experiment 1 (and 1*) for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
1	48.5	46.26	0.44	10.15	13.35	0.0	7.51	8.76	11.83
1*	45.73	44.51	0.35	8.09	9.57	0.0	4.74	6.8	9.04

The KPIs for Experiment 1 are shown in Table 9.5. Again, a systemic overestimation is observed ($ME = 7.69$), which is covered by the two-sided t-test. Additionally, a poorer performance in 2019 compared to the remaining data set is observed. The comparison of Experiment 0* and 1 (Figure 9.9) unveils that the decrease in estimation accuracy when the route is drawn is negligible, since the MAE only increases by 0.52 (Experiment 0*: 8.42). This is explainable with the characteristics of the routes, as shown in Chapter 7. Since in most cases the most used route is used more than 90% of the time, deviations from this route are unlikely. It is worth mentioning that the comparison of Experiment 0* (known route) with Experiment 1 (route draw) also functions as a comparison of the route approximators ARA (active routes) with HRA (historic routes). Thus, we conclude that the route characteristics in the given case allow us to use HRA without harming the estimation accuracy.

When adding the knowledge of the stage start times – which is expressed through Experiment 0 – the MAE is reduced from 8.94 for Experiment 1 to 6.44 for Experiment 0, which is a drop of nearly 28%. This knowledge helps to prevent a sum up of estimation errors, leading to shifted operation start times and therefore incorrect look ups of feature values.

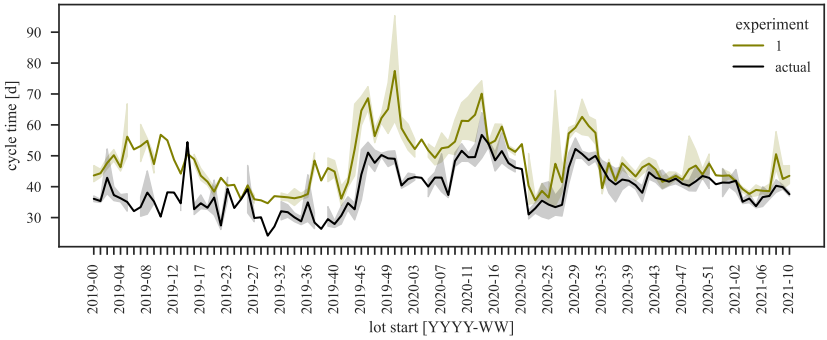
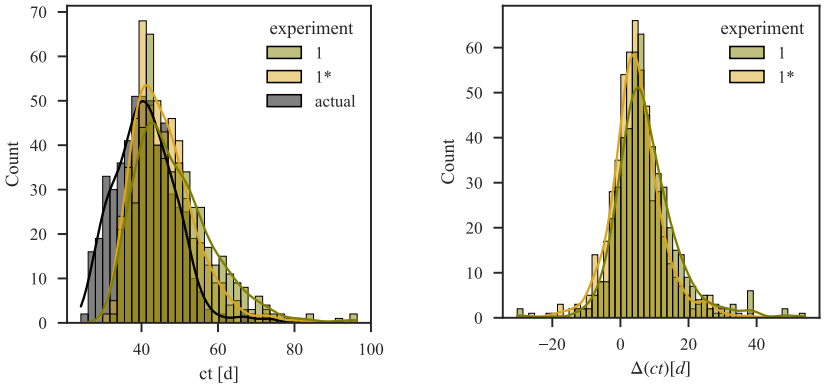


Figure 9.7: Average cycle time over time for Experiment 1 and the actual cycle time.

Similar to Experiment 0, it is possible to adjust Experiment 1 so that the stage start times are known, which we call **Experiment 1***. This enriches the analysis with the impact of the route methodology on the performance of the methodology. First, Experiment 1 and 1* are compared as visualized in Figure 9.8.



(a) Absolute

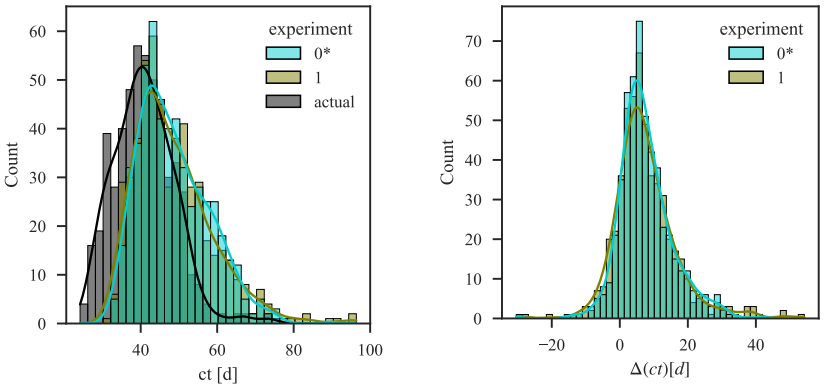
(b) Relative

Figure 9.8: Distribution of predictions for Experiment 1 and Experiment 1 when the stage start times are known (Experiment 1*)

Similar to Experiment 0, we observe a reduction of the residual variability when including the knowledge of the stage start times. This is to be expected, since known stage start times reduce the probability of increased start time deviations for operations at the end of the route, leading to heavier sum ups of estimation errors.

Especially the performance of Experiment 1* for lots with high cycle time, as shown in Table D.5, is worth mentioning, as the $RMSE$ is the lowest of all experiments. Hence, the error of outlier lots in this experiment is rather low, and its mean (52.92) nearly exactly matches the actual mean ($\bar{X} = 52.94$), resulting in an ME of only -0.02. This finding is underlined by $p = 0.9883$, which indicates

that the hypothesis that Experiment 1* and the actual cycle time distribution share the same mean for lots with high cycle times can not be rejected.



(a) Absolute **(b)** Relative

Figure 9.9: Distribution of predictions for Experiment 0* and 1.

9.6 Experiments 2-5: Impact of feature uncertainty

By including feature estimation, these experiments conclusively evaluate how the methodology performs under the effect of input feature estimation uncertainty. The strategies developed in Section 6.2.2 are used in the following.

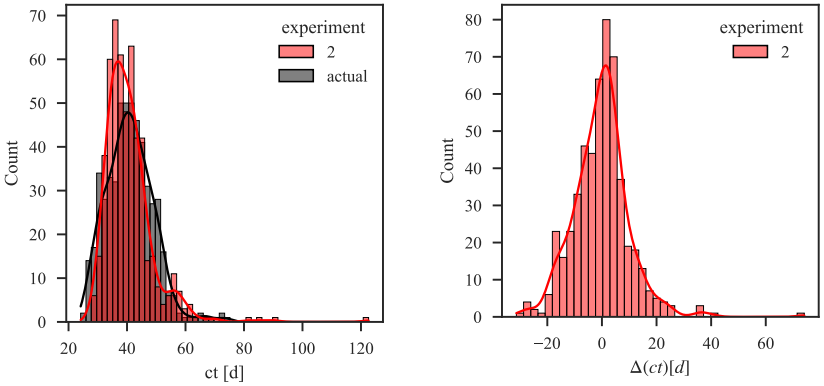
9.6.1 Experiment 2: Median feature values in a fixed time window at lot start

The design of the experiment for this setting is presented in Figure 8.1.

As mentioned, the time window which is taken for the median calculation can have an effect on the performance. Therefore, the calculation of cycle times is executed with different time windows. The performance measures introduced in Section 2.4 are calculated for several time windows, as shown in Table 9.6. A time window of four months is chosen, because the resulting MAE (7.57) and $RMSE$ (10.42) are the lowest and it's ME is close to 0, which indicates no heavy systematical shift in the predictions.

Table 9.6: Key performance indicators for time window comparison for experiment 2 in days.

time window	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	ME	MAE	$RMSE$
10 hours	43.71	41.72	0.41	9.33	2.72	8.66	11.63
1 week	46.12	40.45	1.38	31.7	5.13	12.28	32.96
2 weeks	51.13	40.61	1.94	44.41	10.15	17.24	46.32
1 month	44.12	40.76	0.58	13.22	3.13	10.29	15.53
2 months	41.94	39.92	0.4	9.11	0.95	8.00	11.3
3 months	41.14	39.81	0.35	8.11	0.15	7.66	10.47
4 months	41.03	39.53	0.37	8.46	0.04	7.57	10.42
5 months	41.02	39.12	0.39	8.85	0.03	8.13	11.11
6 months	41.30	39.03	0.42	9.51	0.31	8.69	11.92
1 year	41.75	39.15	0.40	9.08	0.77	9.10	12.10
2 years	42.88	40.23	0.57	13.02	1.89	9.82	15.58



(a) Absolute

(b) Relative

Figure 9.10: Comparison of cycle time predictions for Experiment 2

Table 9.7: Key performance metrics for Experiment 2 for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
2	40.71	39.05	0.37	8.5	-0.55	0.58	-0.28	7.37	10.26

Table 9.8: Key performance metrics for Experiment 2 for all validation lots with a cycle time lower than 35 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	31.18	31.35	0.23	2.53					
2	36.34	35.41	0.52	5.78	9.15	0.0	5.16	5.5	7.77

Table 9.9: Key performance metrics for Experiment 2 for all validation lots with a cycle time higher than 48 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	52.94	51.24	0.58	5.63					
2	41.73	38.95	1.29	12.53	-7.91	0.0	-11.21	14.96	17.58

With the chosen time window, the distribution of predicted cycle times is visualized in Figure 9.10. As shown in Table 9.7, with an MAE of 7.57 and an ME of 0.04, nearly no systematical bias is observed. Consequently, the resulting p of the two-sided t-test is significantly higher than for all previous experiments, and the hypothesis that Experiment 2 and the actual cycle times share the same mean can not be neglected. As expected, the width of the distribution is more narrow compared to the actual distribution, since the feature estimation is based on the average from a fixed time window, which does not allow feature variation throughout the lot execution. Interestingly, there are still predictions with cycle times over 80 days. We investigate the route choices from these lots and observe that they have taken unusual routes, indicating a much longer stay in the fab than actually realized. In reality, such lots can be accelerated by being given a higher priority when they are late, but since this is not covered in this methodology, as explained in Section 6.2.2, those long cycle time predictions occur.

Analyzing lots with cycle times off the mean, as shown in Table 9.8 and Table 9.9, it shows that the configuration of Experiment 2 (low cycle time lots: $MAE = 6.09$; high cycle time lots: $MAE = 14.96$) improves the accuracy of prediction for lots with low cycle times (it is the only experiment where MAE is reduced to 6.09 compared to the overall MAE of 7.57), while it heavily underestimates the lots with high cycle times. The ME for lots with high cycle times is -11.21, indicating a systematical error of over 11 days. Consequently, the configuration in Experiment 2 seems to be unable to identify outlier lots properly. This can be explained with the characteristics of Experiment 2, since the median feature usage in a fixed time window does not allow strong variation in the input features.

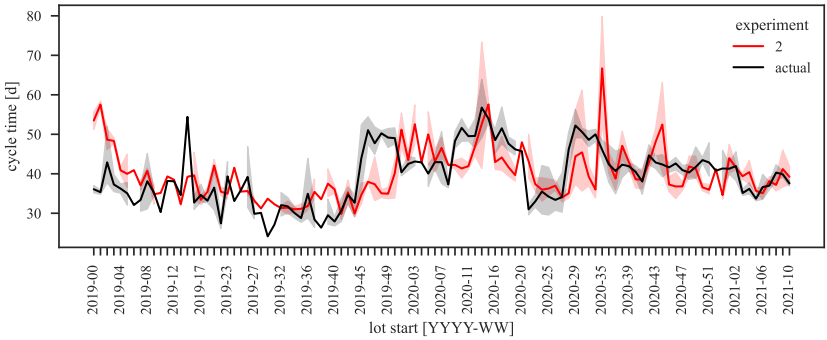


Figure 9.11: Average cycle time over time for Experiment 2 and the actual cycle time.

The development of the estimated cycle time compared to the actual cycle time is shown in Figure 9.11. Especially compared to Experiment 0, the estimations do not follow seasonal and overall trends comparably, but have a tendency to be closer to the mean, which is again explainable by the mechanics of this experiment.

9.6.2 Experiment 3: Median feature values in rolling time window

As in Experiment 2, the first target is to define the time window, which also remains true when a rolling average approach is used. Therefore, the experiment is executed with the same time windows. The detailed optimization of the time window is explained in Appendix B. The optimal time window seems to be – like in Experiment 2 – four months. This is expected since the time window mechanism is similar. Experiment 3 is more accurate but the general relationship between the length of the time window and the performance remains the same.

Table 9.10: Key performance metrics for Experiment 3 for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
3	40.82	39.99	0.31	7.01	-0.35	0.72	-0.16	5.27	7.59

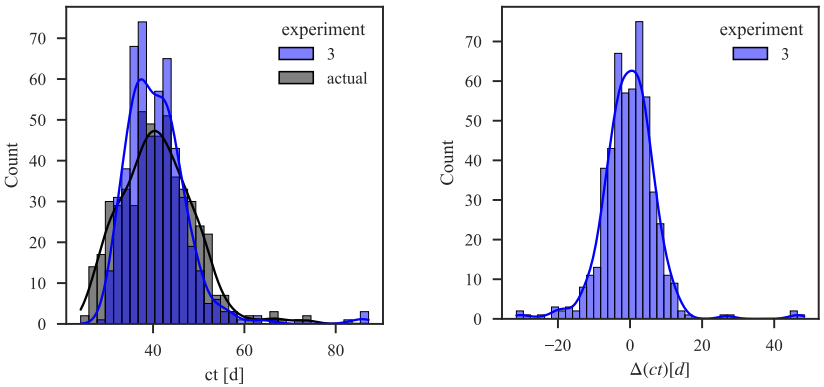
Table 9.11: Key performance metrics for Experiment 3 for all validation lots with a cycle time lower than 35 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	31.18	31.35	0.23	2.53					
3	35.64	35.51	0.32	3.59	11.34	0.0	4.46	4.61	5.63

Table 9.12: Key performance metrics for Experiment 3 for all validation lots with a cycle time higher than 48 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	52.94	51.24	0.58	5.63					
3	46.29	44.93	0.74	7.15	-7.08	0.0	-6.65	8.48	11.0

With the chosen time window, the distribution of predicted cycle times is visualized in Figure 9.12, while the resulting KPIs are shown in Table 9.10. The null hypothesis that the underlying distribution mean of Experiment 3 and the actual cycle time distribution are identical, can not be rejected. Thus, the estimation has no significant bias, which is also reflected in the mean error ($ME = -0.29$). Experiment 3 results in a MAE of 5.15, which is worth noticing since this is even lower than the observed MAE of the benchmark (Experiment 0: 6.44). It is additionally notable that Experiment 3 seems to perform similarly well for outlier lots. For lots with low cycle time an MAE of 5.49 is observed (see Table 9.11), which is the lowest value of all experiments. Also for lots with high cycle time, as shown in Table 9.12, an MAE of 8.42 is the best result.



(a) Absolute

(b) Relative

Figure 9.12: Comparison of cycle time predictions for Experiment 3

The time-related development of the cycle time using Experiment 3 compared to the actual cycle time development is visualized in Figure 9.13. The experiment is capable to follow the seasonal and overall trends better than Experiment 2. The comparison of both experiments regarding cycle time development would exceed the scope of this thesis, but is done in Appendix C. The visualization shows that the methodology also in the configuration of Experiment 3 underestimates the cycle time in phases of high average cycle time, and overestimates it in phases of low average cycle time. Since the given data set covers more phases of low cycle time instead of high cycle time, the ME is positive.

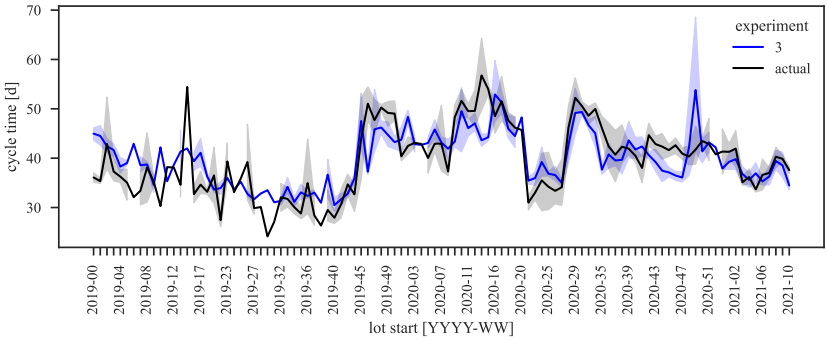
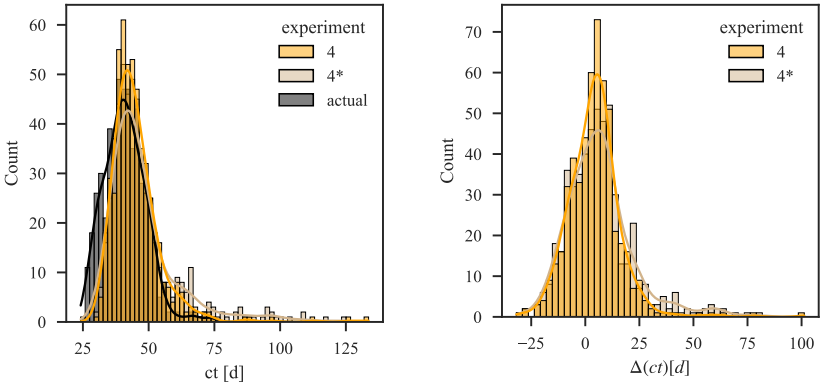


Figure 9.13: Average cycle time over time for Experiment 3 and the actual cycle time.

9.6.3 Experiment 4: Draw of feature values in a fixed time window

While the methodology configurations used in Experiment 2 and 3 provide are expected to suffer from underestimating variations, due to averaging in the input feature estimation, the draw of feature values from their distribution induces more randomness to the input parameters. This can even be increased when the features are allowed to be drawn from any time point (setting the time window to the duration of the entire data set). Since the cycle time varies significantly in the given data set, a random draw of values potentially leads to a bias in cycle time estimation. Therefore, a time window is introduced, in which the algorithm is allowed to draw values. Similar to Experiment 2 and 3, we evaluate the optimal time window, using the same methodology. We provide a more detailed analysis in Appendix B. A very short time window such as 10 hours is chosen. Hence, this will be used to evaluate the methodology in the following. It is worth noticing that a small time window seems to be optimal. A possible explanation could be that feature draws in long time windows open the possibility for impossible feature combinations as well as a bias of feature values near the overall mean. When a short time window is chosen, only a few lots and fab situations are taken into consideration, which possibly reduces this risk, while the risk of not

including enough comparable lots is increased. Since it is expected that cycle times of lots are autocorrelated with their predecessors, this risk seems to be negligible. When applied, the distribution visible in Figure 9.14 can be realized, while the KPIs are shown in Table 9.13. It gets visible that the methodology with the configuration of Experiment 4 tends to systematically overestimate the cycle time, which is indicated by an ME of 4.51. Accordingly, the null hypothesis of identical distribution means of Experiment 4 and the actual cycle time can be rejected, either for all validation lots as well as for lots with low or high cycle time. An explanation for this behavior is that this configuration comes along with a higher risk to draw a high outlier than a low outlier, because of the long right tail of the cycle time distribution. Additionally, the variance of the estimated distribution σ_X is significantly higher than the actual distribution, which indicates that a random draw of feature values induces too much randomness into the system.



(a) Absolute **(b)** Relative

Figure 9.14: Comparison of cycle time predictions for Experiment 4 and 4*.

Table 9.13: Key performance metrics for Experiment 4 (and 4*) for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
4	45.31	43.41	0.45	10.24	7.65	0.0	4.33	9.35	13.11
4*	47.7	44.03	0.6	13.63	9.74	0.0	6.71	12.03	17.08

Table 9.14: Key performance metrics for Experiment 4 (and 4*) for all validation lots with a cycle time lower than 35 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	31.18	31.35	0.23	2.53					
4	42.65	40.12	1.01	11.33	11.04	0.0	11.47	11.51	16.25
4*	44.33	40.65	1.02	11.46	12.53	0.0	13.15	13.27	17.1

Table 9.15: Key performance metrics for Experiment 4 (and 4*) for all validation lots with a cycle time higher than 48 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	52.94	51.24	0.58	5.63					
4	44.2	42.07	1.3	12.62	-6.13	0.0	-8.74	11.92	15.97
4*	43.42	41.88	1.34	12.97	-6.52	0.0	-9.51	12.93	16.32

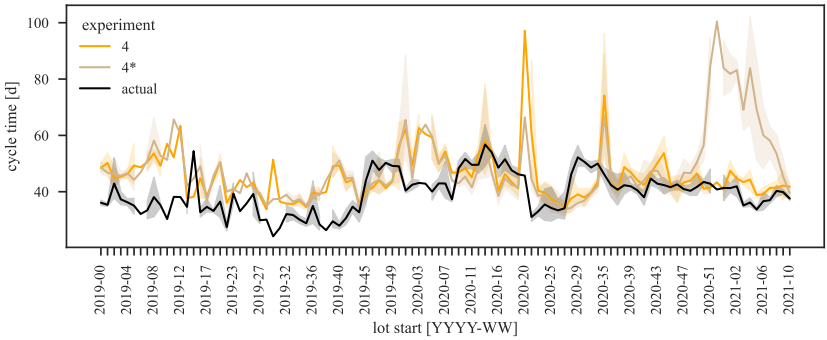


Figure 9.15: Average cycle time over time for Experiment 4 and the actual cycle time.

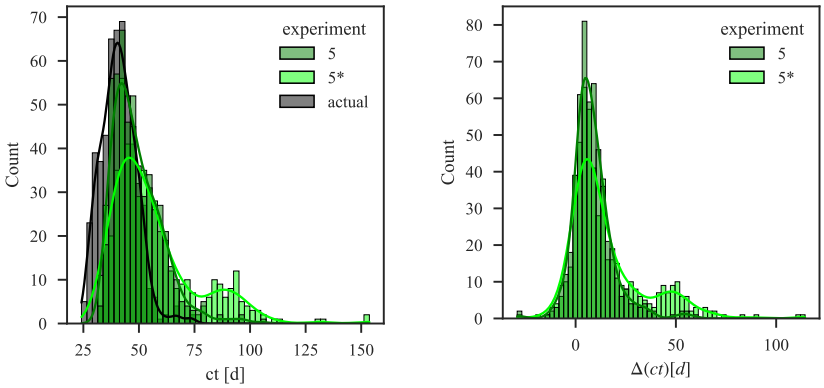
The comparison of the average estimated cycle time using Experiment 4 and the actual cycle time development is visualized in Figure 9.15. It demonstrates that Experiment 4 tends to produce outliers in some weeks, combined with a tendency not to follow trends in the data very well, which can again be explained with the problems of random feature value draws mentioned before.

As mentioned previously, it is also possible to draw feature values all together, by drawing example lots and using their feature vector to ensure sensible feature combinations. The optimal time window for this configuration, which we call **Experiment 4*** is again examined using the methodology introduced for Experiment 2. Its details are also shown in Appendix B. The resulting time window is again 10 hours, which we use in the following.

As shown in Table 9.13, Table 9.14 and Table 9.15, the methodology in the configuration of Experiment 4* does not perform any better than Experiment 4. While the estimations for lots with low cycle time are on average two days more accurately, estimations for all validation lots are on average 2.5 days and for lots with high cycle time on average 1 day less accurately. Therefore, the draw of feature values all together does not seem to have a beneficial impact on the estimation accuracy.

9.6.4 Experiment 5: The draw of feature values in a rolling time window

The final experiment aims to reveal the performance when a feature value is drawn in a rolling time window. As for all realistic experiments, the optimal time window has to be identified first, which is again executed according to the methodology explained in detail for Experiment 2. The details of the time window optimization for this experiment are visible in Appendix B. Again, a time window of 10 hours is used in the following.



(a) Absolute **(b)** Relative

Figure 9.16: Comparison of cycle time predictions for Experiment 5 and 5*.

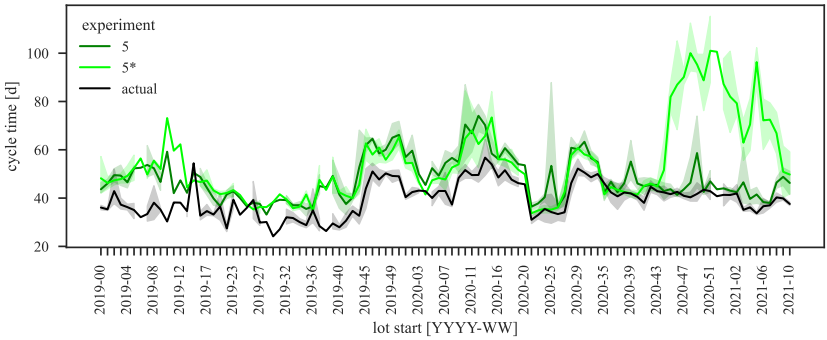


Figure 9.17: Average cycle time over time for Experiment 5, 5* and the actual cycle time.

With the chosen time window, the distributions visible in Figure 9.16 are realized. The corresponding KPIs are shown in Table 9.16, Table 9.17 and Table 9.18. Again, the null hypothesis of identical distribution means for Experiment 5 and actual can be rejected. It can be observed that the predictions have an underlying systematic positive shift in the estimated cycle time compared to the actual distribution, since a ME of 8.16 is realized. It is worth mentioning that the null hypothesis of similar distribution means for Experiment 5 and the actual cycle time is rejected, which is also reflected by a positive ME (6.35) for lots with high cycle time. Consequently, the predictions tend to be slightly overestimated. The comparison of the average estimated cycle time using Experiment 5 and the actual cycle time development is shown in Figure 9.17. As visible, Experiment 5 tends to generally overestimate the average cycle time.

Table 9.16: Key performance metrics for Experiment 5 for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
5	49.29	47.13	0.49	11.15	13.89	0.0	8.31	9.47	12.95
5*	56.19	50.94	0.83	18.98	16.92	0.0	15.21	16.18	24.09

Table 9.17: Key performance metrics for Experiment 5 (and 5*) for all validation lots with a cycle time lower than 35 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	31.18	31.35	0.23	2.53					
5	42.38	39.6	0.71	7.97	14.96	0.0	11.19	11.19	13.87
5*	45.73	42.04	1.24	13.86	11.55	0.0	14.55	14.55	19.72

Table 9.18: Key performance metrics for Experiment 5 (and 5*) for all validation lots with a cycle time higher than 48 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	52.94	51.24	0.58	5.63					
5	59.28	58.97	0.98	9.54	5.55	0.0	6.35	9.98	12.58
5*	58.91	56.61	1.13	10.91	4.72	0.0	5.98	8.97	12.87

As already mentioned, it is also possible to draw the feature values all together, by iteratively choosing a lot in a defined time window and taking the feature values from this time point. The optimal time window for this configuration, which we call **Experiment 5***, is also 10 hours, which is again determined using the process explained in detail for Experiment 2. The detailed analysis exceeds the scope of this thesis, but is provided in Appendix B.

As shown in Table 9.16, Experiment 5* performs similarly compared to Experiment 5. While it seems to estimate lots with high cycle times on average approx. 1.3 days more accurate, as shown in Table 9.18, its estimations for all validation lots are less accurate both for all validation lots and for lots with low cycle times, as shown in Table 9.17.

9.6.5 Comparison of Experiment 2-5

Table 9.19: Key performance metrics for Experiments 0, 1, 2, 3, 4 and 5 for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
0	45.75	44.33	0.36	8.33	9.47	0.0	4.76	6.38	8.45
0*	48.6	47.17	0.39	9.04	14.49	0.0	7.61	8.42	10.86
1	48.5	46.26	0.44	10.15	13.35	0.0	7.51	8.76	11.83
1*	45.73	44.51	0.35	8.09	9.57	0.0	4.74	6.8	9.04
2	40.71	39.05	0.37	8.5	-0.55	0.58	-0.28	7.37	10.26
3	40.82	39.99	0.31	7.01	-0.35	0.72	-0.16	5.27	7.59
4	45.31	43.41	0.45	10.24	7.65	0.0	4.33	9.35	13.11
4*	47.7	44.03	0.6	13.63	9.74	0.0	6.71	12.03	17.08
5	49.29	47.13	0.49	11.15	13.89	0.0	8.31	9.47	12.95
5*	56.19	50.94	0.83	18.98	16.92	0.0	15.21	16.18	24.09

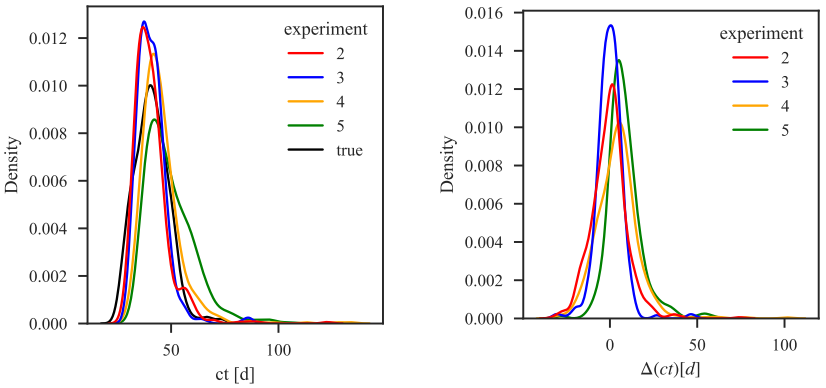
Table 9.20: p-values of one-sided t-test for Experiments 0, 1, 2, 3, 4 and 5 with each other with H_0: MAE of Experiment x is greater or equal the MAE of Experiment y for all validation lots.

		y									
		0	0*	1	1*	2	3	4	4*	5	5*
x	0	0.5	0	0	0.12	0.01	1	0	0	0	0
	0*	1	0.5	0.23	1	0.99	1	0.03	0	0.02	0
	1	1	0.77	0.5	1	1	1	0.13	0	0.09	0
	1*	0.88	0	0	0.5	0.08	1	0	0	0	0
	2	0.99	0.01	0	0.92	0.5	1	0	0	0	0
	3	0	0	0	0	0	0.5	0	0	0	0
	4	1	0.97	0.87	1	1	1	0.5	0	0.41	0
	4*	1	1	1	1	1	1	1	0.5	1	0
	5	1	0.98	0.91	1	1	1	0.59	0	0.5	0
	5*	1	1	1	1	1	1	1	1	1	0.5

Figure 9.18 shows the histograms of cycle times for Experiment 2 to 5, while Table 9.19 shows the corresponding KPIs. The comparison of the average cycle time estimation over time for those experiments is shown in Appendix C for the interested reader. The methodology with the configurations of Experiment 4 and 5 result in higher ME and MAE than of Experiment 2 and 3. Additionally, the p-values for the one-sided t-test for every combination of experiments are shown in Table 9.20. The corresponding test statistic values are shown in Table D.2 for the interested reader. As visible, the null hypothesis of higher or equal MAE of Experiment 3 (x) can be rejected for all other experiments. Additionally, the same hypothesis can be rejected for Experiment 2 when comparing to all other Experiment with all degrees of freedom (4, 4*, 5, 5*). Thus, a certain amount of input feature smoothing seems to be beneficial for the estimation quality.

Furthermore, it is notable that the configuration of the methodology as in Experiment 3 performs as well as in Experiment 0 and 1. Even though its MAE is even lower than the MAE of Experiment 0 and 1, and also the one-sided t-test

indicates a significantly reduced MAE, it is not valid to conclude a superiority of Experiment 3 over Experiment 0 and 1. The rejection of the two-sided t-test null hypothesis for Experiment 0 and 1 compared to the fact that the null hypothesis could not be rejected for Experiment 3 indicate that the lower MAE mainly stems from a reduced bias. The tendency to underestimate is immanent in the configurations of Experiment 2 and 3, because the feature input smoothing reduces the impact of outliers. Thus, those configurations seem to profit from the nature of the given data set with several heavy outliers, while Experiment 0 and 1 incorporate those outliers more. Hence, it seems that the estimation with median values is beneficial for the given case, which can not be generalized for other data sets. These findings suggest that improvement of the feature set that the models may use will have higher impact on estimation performance than the observed difference in feature or route estimation methods.



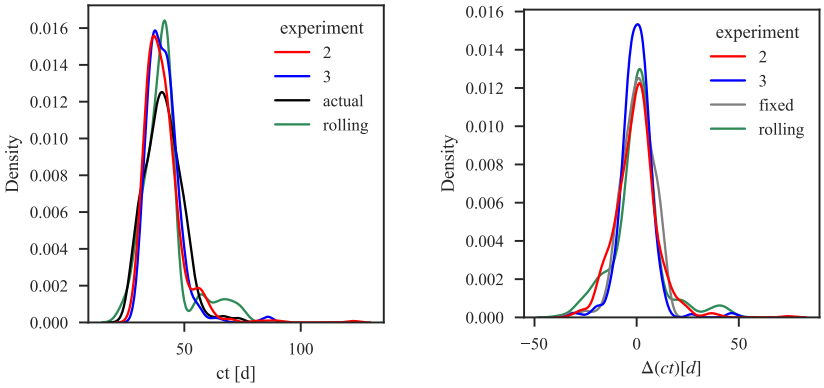
(a) Absolute **(b)** Relative

Figure 9.18: Absolute and relative density approximation of the histogram of cycle time for all experiments that include both route and feature uncertainty.

In summary, the methodology in the configuration of Experiment 3 provides the best results in terms of the MAE and $RMSE$. Additionally, the findings of the t-test suggest that Experiment 3 is the only Experiment with no significant

bias. Therefore, it is further used in the following comparison with the standard approaches commonly in use in industry applications. Nonetheless, since – as mentioned in Section 8.2.2 – a connection of a fab simulation is needed for the execution of the configuration of Experiment 3, also Experiment 2 will be further analyzed.

9.7 Comparison of optimal configuration of methodology with industrial standard approaches



(a) Absolute **(b)** Relative

Figure 9.19: Absolute and relative density approximation of the histogram of cycle time of Experiment 2, 3 and the industrial standard approaches. In Subfigure (a) the fixed approach was removed to increase the visibility.

The approaches for long-term estimations commonly used in the industry are described in Section 4.2. We replicate these approaches and apply them to the data set described in Section 9.2. Two approaches are replicated: First a **fixed** mean, and second a **rolling** mean. In the following, we compare them to

the methodology developed in this thesis. Since we focus on a comparison of models that are usable with unknown routes and feature values, we only compare the results from Experiment 2 and 3 with the ones from the commonly used approaches in industry. Their distribution and the actual distribution is shown in Figure 9.19 (in Figure 9.19 (a) we remove the "fixed" approach, because it contains by definition no variation and therefore deteriorates the visibility of all other distributions within the plot).

Table 9.21: Key performance metrics for Experiment 2 and 3 and the industrial standard approaches for all validation lots (t-test of the fixed mean approach is not indicated, because the values are not normally distributed, which is an underlying assumption for the correct application of the test).

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
2	40.71	39.05	0.37	8.5	-0.55	0.58	-0.28	7.37	10.26
3	40.82	39.99	0.31	7.01	-0.35	0.72	-0.16	5.27	7.59
fixed	41.3	41.3	0.0	0.0			0.31	6.24	7.93
rolling	41.46	41.09	0.43	9.89	0.85	0.4	0.47	8.16	12.38

The industrial standard approaches visibly tend to estimate values around the mean (the fixed mean approach always uses the observed mean over a recent time window as a prediction), while the variation of the cycle time is not covered appropriately. This is to be expected since these approaches does not cover the route variation or any elaborated version of feature dependency on the cycle time, as underlined by the KPIs in Table 9.21. Note that $\bar{X}$ of the actual distribution does not equal $\bar{X}$ of the fixed mean, because the fixed mean approach considers all lots, while the actual distribution only considers validation lots. This also implies that – as visualized in Figure 9.20 – the industrial standard approach "rolling" does not follow the actual development with a defined latency. Nevertheless, it has to be mentioned that the "rolling" approach does follow the trends in the data, but is prone to outliers, which results in heavy variation in some parts of the plot. As expected, the "fixed" approach does not vary at all.

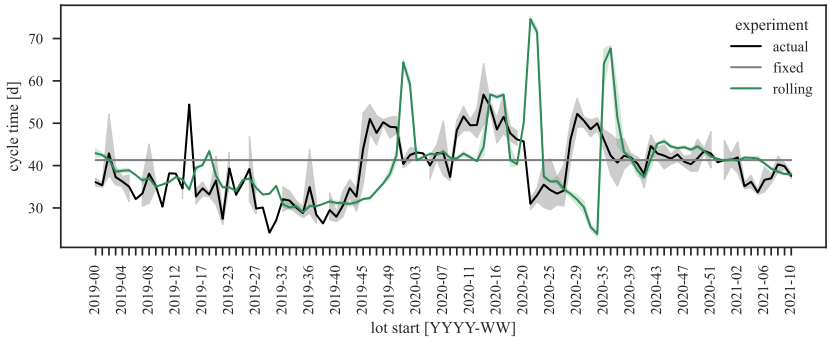


Figure 9.20: Average cycle time of validation lots over time for the industrial benchmark approaches and the actual cycle time. The deviations of the rolling mean approach from the true values stem from the train-test-validation split. Thus, those high deviations are due to the inclusion of *all* lots in the rolling mean approach (as in reality).

Table 9.22: p-values of one-sided t-test for Experiments 2, 3 and the industrial standard approaches with each other with H_0: MAE of Experiment/approach x is greater or equal the MAE of Experiment/approach y for all validation lots.

		y			
		2	3	fixed	rolling
x	2	0.5	1	1	0.06
	3	0	0.5	0	0
	fixed	0	1	0.5	0
	rolling	0.94	1	1	0.5

The p-values of the one-sided t-test, as shown in Table 9.22 (test statistic shown in Table D.10), indicates that the hypothesis of higher or equal MAE values in Experiment 3 compared to the industrial standard approaches can be rejected, while higher MAE values of Experiment 2 compared to the fixed mean have to be assumed. Nonetheless, when looking at the KPIs, it is remarkable that there is no significant improvement of MAE of Experiment 3 compared to the industrial standard approaches. This is expectable, since the industrial standard

approaches are by definition a proper way to estimate mean values. Hence, in the mean, lots are predicted well using those approaches. This is especially notable for the "fixed" mean approach, which is performing nearly as well as Experiment 3. The performance difference between the industrial standard approaches and Experiments 2 and 3 becomes more significant when focusing on those lots which deviate from the mean. This analysis is presented in the following.

Table 9.23: Key performance metrics for Experiment 2 and 3 and the industrial standard approaches for all validation lots with a cycle time lower than 35 days (t-test of the fixed mean approach is not indicated, because the values are not normally distributed, which is an underlying assumption for the correct application of the test).

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	31.18	31.35	0.23	2.53					
2	36.34	35.41	0.52	5.78	9.15	0.0	5.16	5.5	7.77
3	35.64	35.51	0.32	3.59	11.34	0.0	4.46	4.61	5.63
fixed	41.3	41.3	0.0	0.0			10.11	10.11	10.42
rolling	40.3	35.96	1.2	13.43	7.46	0.0	9.12	9.61	15.89

Table 9.24: p-values of one-sided t-test for Experiments 2, 3 and the industrial standard approaches with each other with H_0: MAE of Experiment/approach x is greater or equal the MAE of Experiment/approach y for all validation lots with a cycle time lower than 35 days.

		y			
		2	3	fixed	rolling
x	2	0.5	0.94	0	0
	3	0.06	0.5	0	0
	fixed	1	1	0.5	0.67
	rolling	1	1	0.33	0.5

For this analysis, the KPIs are calculated for outlier lots. In this case, outlier means lots with very low cycle times ($\leq$ 35 days) as well as lots with very high cycle times ($\geq$ 48 days). The results for all lots with low cycle times are shown

in Table 9.23. The corresponding p-values of the one-sided t-test are obtained in Table 9.24 (test statistic shown in Table D.10). In contrast to the industrial standard approaches, the performance of the methodology in the configurations of the experiments hardly deteriorates. The MAE of the rolling mean approach is doubled for those lots compared to all lots, which indicates the problem of such simple approaches: They are not designed to cover strong deviations from the mean. Also the fixed mean approach performs significantly worse for this group of lots. All these findings are covered by the results of the one-sided t-test. The hypothesis that the MAE of one of the industrial standard approaches is equal or higher than the MAE of Experiment 2 or 3 can not be rejected. Therefore we conclude those experiments are superior. This observation is made for any subset of lots. Nonetheless, it is worth mentioning that the fixed mean approach provides a lower MAE than the rolling mean approach. Initially, it was assumed that the rolling mean approach introduces a better variability estimation compared to the fixed mean approach. This assumption was made based on an expected development of the cycle time over a product life cycle, but the real development, which has been shown in Figure 9.1, does not show this expected behavior. This is due to strong variations in cycle time, which have taken place in the course of this data set. Hence, the rolling approach is not capable of coping with the complex and often abrupt cycle time development, while the fixed approach provides in comparison a relatively reliable estimator. It must be noted that this cycle time development is not necessarily typical, since only one data set from one fab is considered. Therefore, the rolling approach is still valid to use as an easy-to-implement method, but it is an interesting outcome of this thesis that this common approach can be outperformed by fixed mean estimation in the current context of serious disruptions in the semiconductor industry.

Table 9.25: Key performance metrics for all experiments for all validation lots with a cycle time higher than 48 days (t-test of the fixed mean approach is not indicated, because the values are not normally distributed, which is an underlying assumption for the correct application of the test).

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	52.94	51.24	0.58	5.63					
2	41.73	38.95	1.29	12.53	-7.91	0.0	-11.21	14.96	17.58
3	46.29	44.93	0.74	7.15	-7.08	0.0	-6.65	8.48	11.0
fixed	41.3	41.3	0.0	0.0			-11.64	11.64	12.91
rolling	39.09	38.96	1.04	10.07	-11.64	0.0	-13.85	15.56	17.44

Table 9.26: p-values of one-sided t-test for Experiments 2, 3 and the industrial standard approaches with each other with H_0: MAE of Experiment/approach x is greater or equal the MAE of Experiment/approach y for all validation lots with a cycle time higher than 48 days.

		y			
		2	3	fixed	rolling
x	2	0.5	1	1	0.32
	3	0	0.5	0	0
	fixed	0	1	0.5	0
	rolling	0.68	1	1	0.5

While this performance gap also can be observed for lots with high cycle times – as shown in Table 9.25 – it becomes clear that also the proposed experiments have bigger problems to estimate the high outliers. Seemingly, those lots are harder to identify based on the input data, which is in line with the struggles production planners have in foreseeing those lots. Nevertheless, Experiment 3 provides a lower MAE than the benchmark approaches, which again is underlined by the p values of the one-sided t-tests. This underlines once more the main advantage of the proposed system, compared to common approaches: It estimates outliers more accurately. In contrast, Experiment 2 provides larger MAE than the fixed

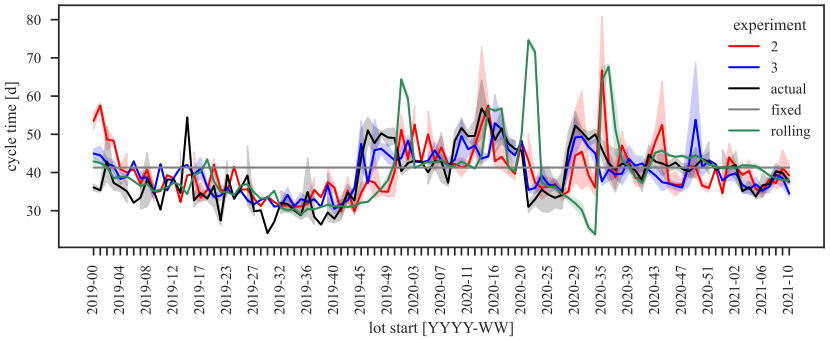


Figure 9.21: Development of the cycle time over all realistic experiments, both industrial standard approaches and the actual development

approach for those lots, which is also indicated by the result of the one-sided t-test.

The development of the cycle time with the different approaches is shown in Figure 9.21. It can be observed that – while the fixed mean approach does not vary at all – the rolling mean approach seems to overestimate certain trends in the cycle time, leading to heavy over- and underestimations in some weeks. Especially Experiment 3 seems to be more robust against such behavior, and is therefore the recommended model configuration. This proofs that the estimating models function, because they are not behaving as a mean approximator in all cases.

Additionally, it can be qualitatively estimated that the proposed methodology has advantages compared to discrete-event-simulation, since it outperforms it in the two areas of disadvantage of simulations described in Section 4.2. First, the proposed method is easier to build and to maintain. Since the methodology is data-driven, every model and sub-methodology works without hard-coding the production processes in it. Additionally, the maintenance is easy to manage, because every operation is modeled independently. Hence, flags can be included in the system, triggering retraining when an operation is adjusted by an engineer, or a certain amount of time has passed. Second, the system is faster than a

simulation. Since no inter-dependencies in queues (except through features such as WIP or queue composition) must be considered, a lot can be calculated without approximating the behavior of all other lots currently in the fab. Hence, calculation times can be drastically decreased from hours to minutes. Since quantitative comparison with a simulation is not in the scope of this thesis, an exact analysis of the calculation time improvement is not conducted.

10 Summary and Outlook

We have developed a concatenated route- and feature-approximating machine learning methodology to estimate cycle times in SC fabs. We have validated this methodology using real operational data from a SC wafer fab of the Robert Bosch GmbH. We have shown that the methodology is capable of outperforming common cycle time estimation approaches such as fixed or rolling means, especially for cycle times deviating from the mean cycle time.

The results of this thesis are achieved through the following three activities, during which additional insights were gained. First, we trained and tuned models for each operation, including an extensive study on the impacting features on operational waiting time. We showed that a variety of features – also newly introduced features that have not been used yet – are necessary to accurately model the full variety of operations, while each operation is only using a fraction of all features.

Second, we implemented route extraction and studied the effect of route variability on our prediction using different route selection methods. Our results indicate that the influence of variable routes on cycle time prediction is smaller than the influence of the other two studied effects, namely the choice and prediction of input variables

Third, we elaborated several approaches dealing with feature uncertainty for estimations in the future. It became evident that a certain amount of input feature smoothing is beneficial for the estimation accuracy. This indicates that future research should rather focus on exceeding and improving the input feature set than improving the uncertainty handling strategies. Additionally, we have shown that a rolling approach – as it could be enabled through the inclusion of a feature simulation – improves the estimation accuracy significantly.

10.1 Discussion

The proposed methodology, which estimates cycle times by concatenating operational sojourn time predictions along a chosen route, shows major advantages to the current state of the art solutions, as described in Section 9.7.

While the development of the methodology as well as the numerical evaluation has shown significant improvement compared to the status quo, we have found some indications that our method is not yet optimal. Thus, we point out aspects where it is still worthwhile to invest further development work in the following. They are discussed for the general methodology as well as for the numerical evaluation separately.

10.1.1 Limitations of the methodology

The developed methodology is a construction of random forests on the base of a chosen route and an estimation of the feature values. Possible areas of improvement are discussed separately for the models themselves, the route selection policy and the feature prediction policy.

Limitations of the waiting time estimation models

The first possible change to discuss is the chosen method. This thesis focuses on random forests, ignoring other options like back propagation networks (BPN) or recurrent neural networks (RNN). We have chosen this modeling technique after an initial analysis, demonstrating that other techniques do not provide better results for the underlying problem. Additionally, the implementation of random forests using the python package sklearn is easy to maintain and therefore reusable in future years. While this is negligible from a scientific perspective, it is considered to be a decision factor, because of the practical background of this thesis. The second area of study is the feature selection. To our knowledge, we have used the most comprehensive feature set for operational waiting time estimation in the literature yet, but we also considered other features which were not available

in our current data set. Especially all features related to transportation time (e.g. utilization of the automated material handling system, distance to the next equipment group) were not available for our data set, but can be expected to improve the prediction power of our models. The third choice worthy of discussion is the scope of the models. Here, we have chosen to build a model for each operation, since this is the smallest entity in the production process. Nevertheless, it could be argued that one model per layer is also a valid option, since they do not have to cope with the route selection problem covered in Chapter 7. If doing so, the feature set must be adapted, since it would include several operations at once. The issue with higher-scope models (stages, layers, the complete cycle time at once) is that all possible features lose impact in long time scopes. Additionally, the modularity of the methodology is lost because retraining is needed for every route and operation change. For this to be worthwhile, higher-granularity models would have to prove that they predict better per process, which may be an interesting study for future research, but is not included in this thesis. Hence, we have chosen a finer granularity in combination with feature estimation strategies.

Limitations of the route selection policy

The introduced route selection methodology contains two major limitations. First, the data driven route identification approach is an approximation of the actual routes. The actual route set is a complex system that follows an if/else logic based on quality and process parameters. A historical data-driven route approximation can only roughly cover the complexity, but is an intentionally chosen simplification, because it is easier to maintain and reduces the complexity of the developed methodology. Second, since no history of active routes is collected in the current use case, the performance of the system in operational use cannot be exactly measured. Therefore, it is currently impossible to tune the route approximation algorithm appropriately. For any further work in this context, it is highly recommended to build a database of active routes, so future solutions can make use of this information. Nonetheless, the set of active routes are available at the point of model execution. Thus, for operational usage instead of historic performance measure, the impact of this limitation is reduced to the point that only the model

training is impacted by it. Therefore, online reinforcement learning techniques are thinkable in the given use case.

Limitations of the feature prediction policies

While the proposed feature approximation strategies provide several options regarding this source of uncertainty, they are by no means a complete set of options. Additionally, it is up to discussion how useful Experiment 3 for the given case actually is, since a simulation is required to estimate feature values on a rolling basis. Since Experiment 3 performs as well as Experiment 0 and 1, the simulation does not necessarily have to be provided with exact feature values, because we showed that smoothing is beneficial. When a fab simulation cannot be provided, it is still preferable to smooth the input features over a median (Experiment 2). This again underlines the immense variability in the input data, which has to be erased partially for accurate predictions. The combination with a simulation has not been further investigated, since it is not in the scope of this thesis. Building a simulation model for a SC fab is a complex and time-intensive task. In comparison, the exploration of machine learning methods as performed in this thesis would have been a minor part. Therefore, we decided to ignore the simulative approach and focus on machine learning, even if our experiments have indicated that a combination of both would be well worth studying to further improve prediction results, as indicated in Experiment 3.

10.1.2 Limitations of the numerical evaluation

We have evaluated the methodology numerically with real operational data of the SC wafer fab from Robert Bosch GmbH, which is by definition realistic. Nevertheless, the considered data set includes a major drop in the demand (2019), a global pandemic (2020), and a sudden and global increase in demand (2021). Even though this increases the value of the data, since it improves the robustness of the system, it is valid to say that the variations in the data set contradict many literature assumptions on the time-development of product cycle times. Since the time frame of this thesis was fixed, there was no real alternative to use the given

data set. Nonetheless, a comparison with other data sets is a worthy direction for future research. Additionally, it has to be mentioned that only one product was used for the evaluation, in order to reduce the dimensionality of the final analysis. Nevertheless, all statements made are not proven to be valid for all products, even if it is likely, since products go through the same production environment, while differing in terms of layers and operations. It is of interest to analyze the performance of the methodology for products with different types or numbers of layers and operations. We expect that a higher number of layers increases the potential variability of the cycle time leading to a competitive advantage of the methodology. By focusing on one high-runner product, another point of criticism can be derived, because the methodology has not been validated with ramp-up products. Thus, it is unknown if estimations for this kind of products are similarly accurate in environments with continuous improvements, which is of potential interest in the future. As mentioned in Section 9.4, the estimations derived in 2019 were significantly less accurate than for 2020 and 2021. Thus, it is a possibility to erase 2019 from the data set, which would likely reduce the estimation error. For machine learning models in operational use, it is common to implement a measurement cycle, where the estimation error is constantly measured and if the error exceeds a certain threshold, retraining with an adjusted data set is triggered. Since the goal of this thesis is to create the methodology for cycle time estimation, such a cycle was not included. Nevertheless, it may be beneficial for the performance and is therefore a potential connection point to further improve the methodology in the future. Finally, one could criticize that the authors have chose a relatively simple set of hypothesis tests, while ignoring other test options such as variance tests (e.g. F-test, Levene-test) or distribution tests (Kolmogorov-Smirnov-test). This limitation was accepted on purpose to keep the analysis intuitive for the interested reader and to not exceed the scope of this thesis. Nonetheless, such tests would contribute to the numerical evaluation and could also be a potential connection point for future research.

10.2 Outlook

The proposed methodology contains two directions for further improvement. First, it is possible to improve the models to estimate the waiting times of operations. This could be done by improving the input data with new or better features or by applying more advanced modeling techniques. As mentioned in Section 9.6.5, exceeding or improving the input feature set seems to be the most promising approach to further improve the results of the methodology. Since this methodology includes automatic feature set reduction, our method is expected to deal well with increasing feature sets, even when redundancies are induced. Second, it is possible to improve the predictability of the features in the future. This could be realized by replacing the proposed and simplified uncertainty handling strategies with a discrete-event simulation. As the experiments have shown, even a simple approximation of the feature values – for example averaging on a weekly basis – improves the performance significantly. Therefore, it is desirable to validate this hypothesis in combination with a simulation. Apart from that, it is desirable to investigate the performance of this methodology in other use cases than a SC wafer fab, as it should be in principle applicable to any possible production configuration, as long as waiting times play a major role in estimating the cycle time. Thus, a study concerning the performance of this methodology for example in SC back ends, automotive or pharmaceutical production environments is of great interest.

A Estimation methods

A.1 Route estimation

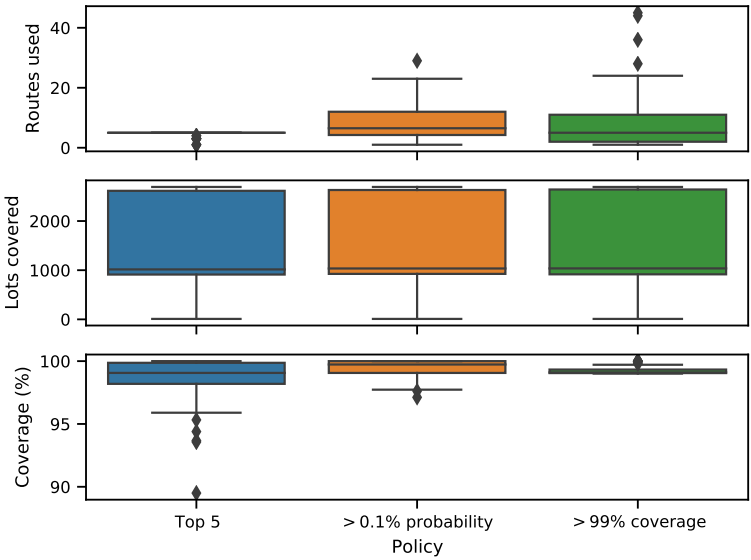


Figure A.1: Impact of the route selection policy on (a) the routes covered, (b) the lots included and (c) the probabilities covered

A.2 Feature estimation

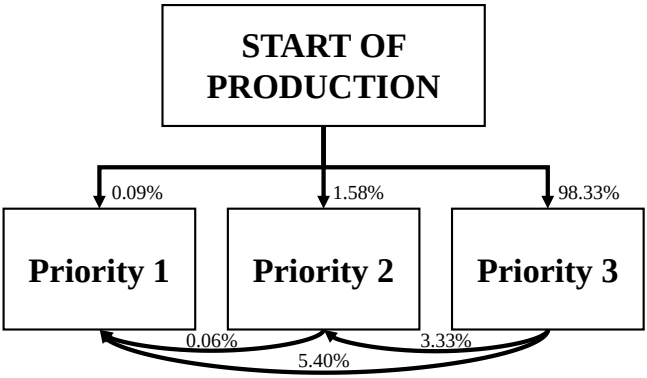


Figure A.2: Entry and Transition probabilities of lot priorities using the data set of the given use case.

B Time Window Experiments

B.1 Experiment 3

As visible in Table B.1, the same tendency of a reduction in the average cycle time estimation can be observed.

Table B.1: Key performance indicators for time window comparison for experiment 3 in days.

time window	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	ME	MAE	$RMSE$
10 hours	46.80	44.88	0.40	9.09	5.81	7.42	10.04
1 week	45.10	42.97	0.38	8.67	4.11	6.37	8.58
2 weeks	44.77	42.28	0.39	8.96	3.79	6.44	8.65
1 month	43.53	42.00	0.33	7.51	2.54	5.95	7.97
2 months	41.89	40.95	0.28	6.47	0.90	5.40	7.53
3 months	40.98	40.42	0.24	5.54	-0.01	5.23	6.97
4 months	40.69	40.26	0.24	5.57	-0.29	5.15	6.77
5 months	40.88	40.48	0.26	5.9	-0.11	5.35	6.99
6 months	40.75	39.96	0.26	6.01	-0.24	5.79	7.69
1 year	41.72	40.52	0.3	6.91	0.73	6.72	8.97
2 years	42.73	41.25	0.29	6.53	1.74	7.03	9.28

B.2 Experiment 4

Table B.2: Key performance indicators for time window comparison for experiment 4 in days.

time window	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	ME	MAE	$RMSE$
10 hours	45.50	43.37	0.46	10.53	4.51	9.55	13.54
1 week	49.92	43.97	1.15	26.39	8.93	13.36	28.89
2 weeks	55.38	45.14	1.53	35.01	14.39	18.55	38.81
1 month	60.05	46.91	1.43	32.70	19.06	22.86	38.59
2 months	61.39	53.64	1.02	23.26	20.4	22.58	31.67
3 months	62.06	60.64	0.85	19.56	21.08	22.57	29.23
4 months	61.85	58.97	0.78	17.94	20.86	21.98	27.59
5 months	62.07	60.51	0.76	17.49	21.08	22.12	27.60
6 months	63.16	60.06	0.78	17.90	22.17	23.26	28.96
1 year	62.34	59.24	0.58	13.28	21.35	22.10	26.19
2 years	61.50	60.08	0.47	10.68	20.51	20.97	24.13

As shown in Table B.2, the width of the errors as well as the mean error correlate with the time window length but stagnate at a time window length of two months. Thus, it seems that the smallest MAE, as well as the least shifted error distribution, can be realized using a short time window like 10 hours.

B.2.1 Experiment 4*

The time window calculation for this sub-experiment is depicted in Table B.3.

Table B.3: Key performance indicators for time window comparison for experiment 4* in days.

time window	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	ME	MAE	$RMSE$
10 hours	44.63	42.99	0.52	9.78	3.8	10.44	13.33
1 week	48.35	43.01	1.21	22.6	7.52	14.08	25.97
2 weeks	54.57	44.34	1.7	31.9	13.74	19.73	36.54
1 month	58.48	47.21	1.46	27.21	17.69	22.97	34.19
2 months	60.52	55.28	1.17	21.87	19.73	22.61	30.92
3 months	59.7	57.25	0.95	17.72	18.9	21.13	26.88
4 months	58.3	57.61	0.83	15.58	17.5	19.57	24.3
5 months	58.41	57.39	0.9	16.8	17.62	19.55	25.38
6 months	55.8	54.07	0.71	13.29	15.01	17.43	22.0
1 year	56.74	55.13	0.75	14.05	15.96	18.08	23.44
2 years	60.0	58.42	0.6	11.21	19.21	20.42	24.69

Since a time window of 10 hours results in the lowest ME and MAE, it is further used.

B.3 Experiment 5

Table B.4: Key performance indicators for time window comparison for experiment 5 in days.

time window	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	ME	MAE	$RMSE$
10 hours	50.42	49.62	0.61	11.35	9.63	10.56	13.69
1 week	51.06	49.29	0.63	11.76	10.26	11.42	14.57
2 weeks	52.59	49.86	0.68	12.74	11.79	12.64	15.38
1 month	54.8	51.5	0.78	14.56	13.99	14.76	17.83
2 months	58.58	55.77	0.84	15.64	17.78	18.5	22.74
3 months	62.77	61.16	0.99	18.45	21.96	22.64	28.1
4 months	62.59	62.06	0.95	17.76	21.79	22.23	27.16
5 months	60.81	58.04	0.91	16.96	20.01	20.98	25.93
6 months	59.9	57.02	0.83	15.4	19.1	19.66	23.96
1 year	59.11	57.36	0.65	12.09	18.3	19.38	22.97
2 years	61.84	60.14	0.58	10.71	21.04	21.53	24.74

As depicted in Table B.4, the same time window with 10 hours as for Experiment 4 is optimal. Again, this is expected since the calculation method within the time window is identical and therefore the overall mechanic remains the same.

B.3.1 Experiment 5*

The KPIs for the time window determination for Experiment 5* is depicted in Table B.5).

Table B.5: Key performance indicators for time window comparison for experiment 5* in days.

time window	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	ME	MAE	$RMSE$
10 hours	49.14	47.17	0.46	10.51	8.16	9.37	12.46
1 week	49.60	47.21	0.43	9.75	8.61	9.5	12.23
2 weeks	50.74	47.87	0.48	10.88	9.75	10.56	13.36
1 month	53.22	49.09	0.61	13.86	12.23	13.17	16.70
2 months	57.20	55.03	0.66	15.11	16.21	16.87	21.37
3 months	60.56	59.09	0.78	17.77	19.57	20.23	26.00
4 months	61.36	59.51	0.71	16.28	20.38	20.76	25.67
5 months	61.76	60.64	0.65	14.96	20.77	21.03	25.08
6 months	62.71	61.30	0.63	14.46	21.73	22.10	25.84
1 year	64.38	62.62	0.50	11.44	23.39	23.67	26.43
2 years	62.40	60.46	0.41	9.35	21.41	21.53	24.03

As visible, a time window of 10 hours minimizes the ME and MAE, and is therefore chosen.

C Cycle Time developments

C.1 Experiment 0

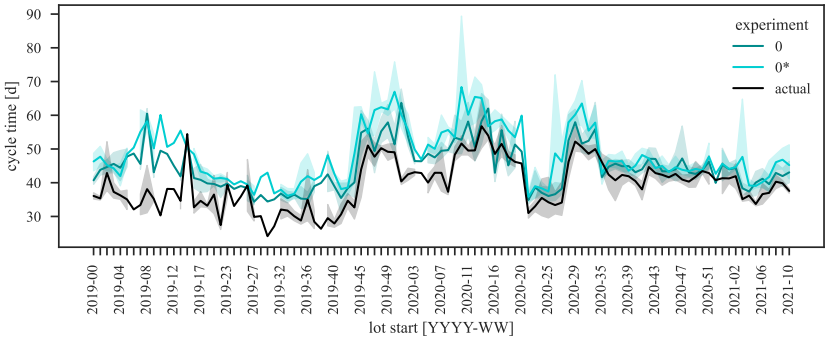


Figure C.1: Average cycle time over time for experiments 0 and 0* and the actual cycle time.

C.2 Experiment 1

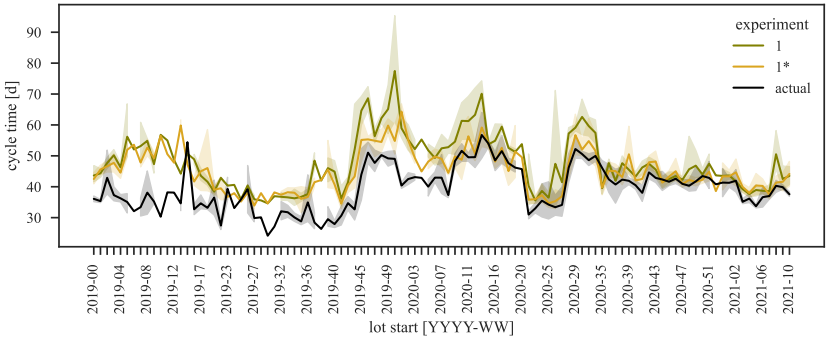


Figure C.2: Average cycle time over time for experiment 1* and the actual cycle time.

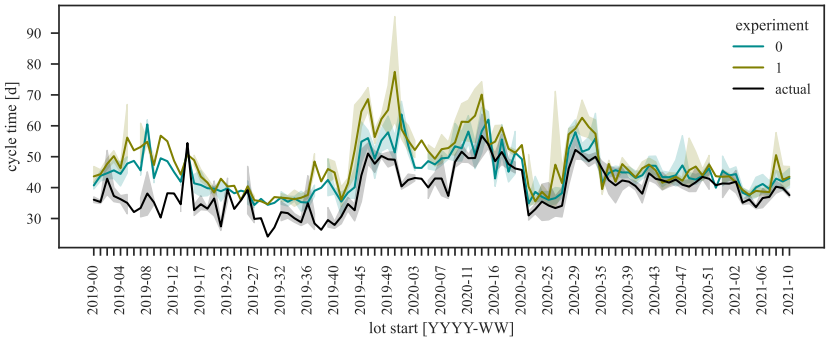


Figure C.3: Average cycle time over time for experiments 0 and 1 and the actual cycle time.

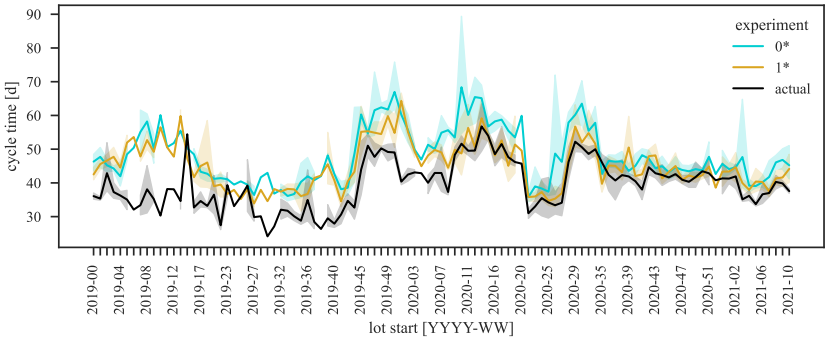


Figure C.4: Average cycle time over time for experiments 0* and 1* and the actual cycle time.

C.3 Experiments 2-5

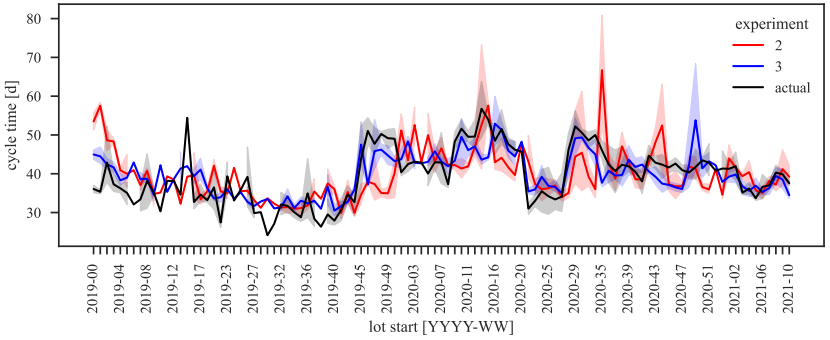


Figure C.5: Average cycle time over time for experiments 2 and 3 and the actual cycle time.

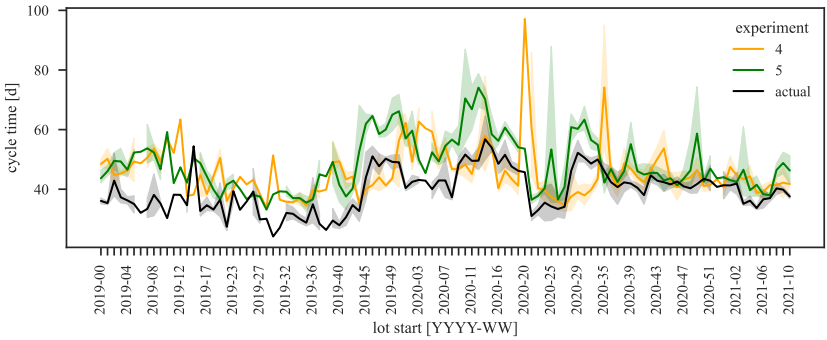


Figure C.6: Average cycle time over time for experiments 4 and 5 and the actual cycle time.

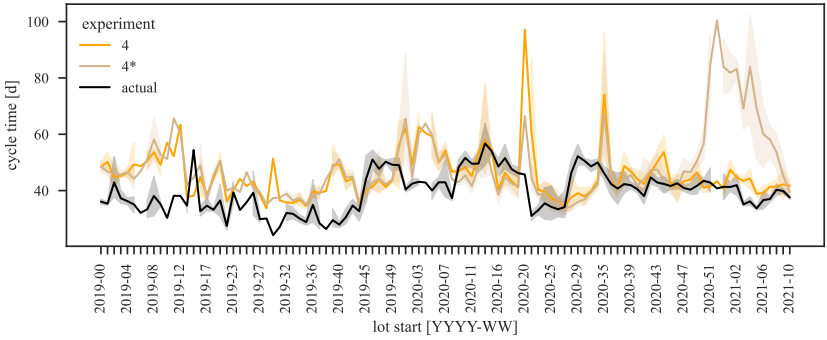


Figure C.7: Average cycle time over time for experiments 4 and 4* and the actual cycle time.

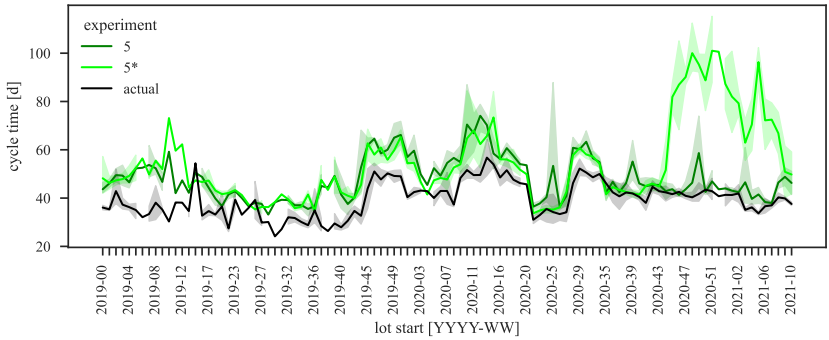


Figure C.8: Average cycle time over time for experiments 5 and 5* and the actual cycle time.

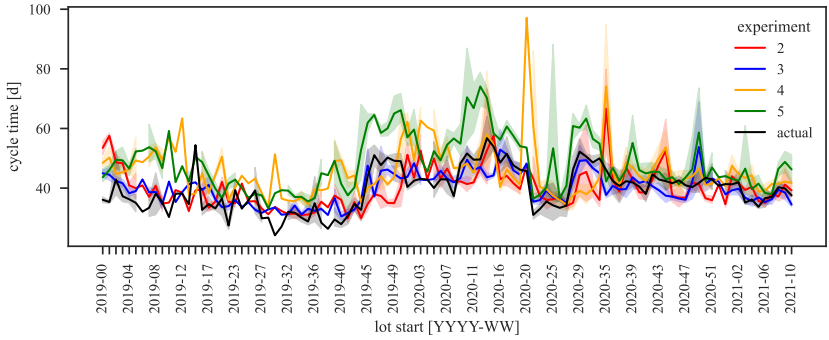


Figure C.9: Average cycle time over time for experiments 2, 3, 4 and 5 and the actual cycle time.

D Numerical evaluation of lots

In the following, all numeric results of the experiments are collected.

Table D.1: Key performance metrics for all experiments for all validation lots.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	40.99	40.59	0.35	7.94					
0	45.75	44.33	0.36	8.33	9.47	0.0	4.76	6.38	8.45
0*	48.6	47.17	0.39	9.04	14.49	0.0	7.61	8.42	10.86
1	48.5	46.26	0.44	10.15	13.35	0.0	7.51	8.76	11.83
1*	45.73	44.51	0.35	8.09	9.57	0.0	4.74	6.8	9.04
2	40.71	39.05	0.37	8.5	-0.55	0.58	-0.28	7.37	10.26
3	40.82	39.99	0.31	7.01	-0.35	0.72	-0.16	5.27	7.59
4	45.31	43.41	0.45	10.24	7.65	0.0	4.33	9.35	13.11
4*	47.7	44.03	0.6	13.63	9.74	0.0	6.71	12.03	17.08
5	49.29	47.13	0.49	11.15	13.89	0.0	8.31	9.47	12.95
5*	56.19	50.94	0.83	18.98	16.92	0.0	15.21	16.18	24.09

Table D.2: Test statistic of one-sided t-test for Experiments 0, 1, 2, 3, 4 and 5 with each other with H_0: MAE of Experiment x is greater or equal the MAE of Experiment y for all validation lots.

		y									
		0	0*	1	1*	2	3	4	4*	5	5*
x	0	0	-4.31	-3.92	-1.69	3.48	5.9	-3.21	-5.01	-3.8	-5.19
	0*	4.31	0	0.15	2.69	7.09	9.31	-0.08	-1.58	0.22	-2.29
	1	3.92	-0.15	0	2.41	6.58	8.53	-0.2	-1.66	0.07	-2.35
	1*	1.69	-2.69	-2.41	0	4.9	7.31	-2.08	-3.77	-2.31	-4.16
	2	-3.48	-7.09	-6.58	-4.9	0	1.57	-5.27	-7.15	-6.43	-7
	3	-5.9	-9.31	-8.53	-7.31	-1.57	0	-6.46	-8.57	-8.33	-8.08
	4	3.21	0.08	0.2	2.08	5.27	6.46	0	-1.24	0.25	-1.92
	4*	5.01	1.58	1.66	3.77	7.15	8.57	1.24	0	1.71	-0.83
	5	3.8	-0.22	-0.07	2.31	6.43	8.33	-0.25	-1.71	0	-2.39
	5*	5.19	2.29	2.35	4.16	7	8.08	1.92	0.83	2.39	0

Table D.3: Key performance metrics for all experiments for all validation lots with a cycle time lower than 35 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	31.18	31.35	0.23	2.53					
0	39.05	37.96	0.47	5.22	15.15	0.0	7.87	7.88	9.46
0*	42.59	40.06	0.66	7.39	16.34	0.0	11.41	11.41	13.64
1	42.43	40.48	0.71	7.97	15.04	0.0	11.25	11.26	13.85
1*	40.27	38.7	0.54	5.98	15.63	0.0	9.09	9.09	10.9
2	36.34	35.41	0.52	5.78	9.15	0.0	5.16	5.5	7.77
3	35.64	35.51	0.32	3.59	11.34	0.0	4.46	4.61	5.63
4	42.65	40.12	1.01	11.33	11.04	0.0	11.47	11.51	16.25
4*	44.33	40.65	1.02	11.46	12.53	0.0	13.15	13.27	17.1
5	42.38	39.6	0.71	7.97	14.96	0.0	11.19	11.19	13.87
5*	45.73	42.04	1.24	13.86	11.55	0.0	14.55	14.55	19.72

Table D.4: Test statistic of one-sided t-test for Experiments 0, 1, 2, 3, 4 and 5 with each other with H_0: MAE of Experiment x is greater or equal the MAE of Experiment y for all validation lots with a cycle time lower than 35 days.

		y									
		0	0*	1	1*	2	3	4	4*	5	5*
x	0	0	-4.31	-3.92	-1.69	3.48	5.9	-3.21	-5.01	-3.8	-5.19
	0*	4.31	0	0.15	2.69	7.09	9.31	-0.08	-1.58	0.22	-2.29
	1	3.92	-0.15	0	2.41	6.58	8.53	-0.2	-1.66	0.07	-2.35
	1*	1.69	-2.69	-2.41	0	4.9	7.31	-2.08	-3.77	-2.31	-4.16
	2	-3.48	-7.09	-6.58	-4.9	0	1.57	-5.27	-7.15	-6.43	-7
	3	-5.9	-9.31	-8.53	-7.31	-1.57	0	-6.46	-8.57	-8.33	-8.08
	4	3.21	0.08	0.2	2.08	5.27	6.46	0	-1.24	0.25	-1.92
	4*	5.01	1.58	1.66	3.77	7.15	8.57	1.24	0	1.71	-0.83
	5	3.8	-0.22	-0.07	2.31	6.43	8.33	-0.25	-1.71	0	-2.39
	5*	5.19	2.29	2.35	4.16	7	8.08	1.92	0.83	2.39	0

Table D.5: Key performance metrics for all experiments for all validation lots with a cycle time higher than 48 days.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	52.94	51.24	0.58	5.63					
0	54.77	54.5	1.0	9.65	1.59	0.11	1.83	8.13	10.81
0*	58.77	58.34	0.79	7.64	5.96	0.0	5.83	8.5	10.87
1	58.33	58.4	0.9	8.74	5.03	0.0	5.39	9.5	11.87
1*	52.92	52.06	0.92	8.9	-0.01	0.99	-0.02	7.48	9.84
2	41.73	38.95	1.29	12.53	-7.91	0.0	-11.21	14.96	17.58
3	46.29	44.93	0.74	7.15	-7.08	0.0	-6.65	8.48	11.0
4	44.2	42.07	1.3	12.62	-6.13	0.0	-8.74	11.92	15.97
4*	43.42	41.88	1.34	12.97	-6.52	0.0	-9.51	12.93	16.32
5	59.28	58.97	0.98	9.54	5.55	0.0	6.35	9.98	12.58
5*	58.91	56.61	1.13	10.91	4.72	0.0	5.98	8.97	12.87

Table D.6: Test statistic of one-sided t-test for Experiments 0, 1, 2, 3, 4 and 5 with each other with H_0: MAE of Experiment x is greater or equal the MAE of Experiment y for all validation lots with a cycle time higher than 48 days.

		y									
		0	0*	1	1*	2	3	4	4*	5	5*
x	0	0	-0.36	-1.31	0.66	-5.65	-0.33	-2.86	-3.78	-1.7	-0.69
	0*	0.36	0	-0.98	1.06	-5.44	0.03	-2.62	-3.54	-1.4	-0.39
	1	1.31	0.98	0	2.04	-4.52	0.99	-1.83	-2.7	-0.44	0.44
	1*	-0.66	-1.06	-2.04	0	-6.43	-1.01	-3.46	-4.44	-2.42	-1.28
	2	5.65	5.44	4.52	6.43	0	5.4	2.08	1.44	4	4.43
	3	0.33	-0.03	-0.99	1.01	-5.4	0	-2.61	-3.52	-1.4	-0.41
	4	2.86	2.62	1.83	3.46	-2.08	2.61	0	-0.67	1.43	2.02
	4*	3.78	3.54	2.7	4.44	-1.44	3.52	0.67	0	2.26	2.81
	5	1.7	1.4	0.44	2.42	-4	1.4	-1.43	-2.26	0	0.81
	5*	0.69	0.39	-0.44	1.28	-4.43	0.41	-2.02	-2.81	-0.81	0

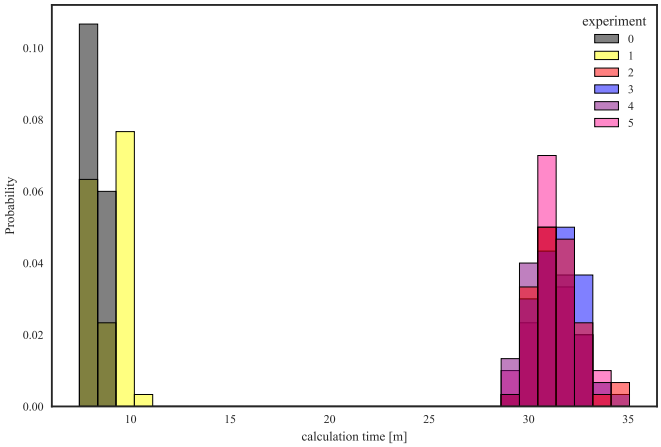


Figure D.1: Histogram of the calculation times of all experiments in minutes.

Table D.7: Key performance metrics for all experiments for all validation lots with a lot start date in 2021.

Exp.	$\bar{X}$	$\widetilde{X}$	$\widehat{\sigma}_{\bar{X}}$	σ_X	t	p	ME	MAE	$RMSE$
actual	38.37	38.1	0.44	3.48					
0	41.62	41.17	0.52	4.07	4.78	0.0	3.25	3.77	4.8
0*	43.49	42.03	0.7	5.52	6.18	0.0	5.12	5.37	7.51
1	42.22	41.2	0.78	6.16	4.28	0.0	3.85	4.54	7.1
1*	41.12	40.42	0.51	3.99	4.09	0.0	2.75	3.55	4.8
2	39.11	37.6	0.6	4.69	0.99	0.32	0.74	3.87	4.96
3	37.31	36.89	0.37	2.91	-1.83	0.07	-1.06	3.24	3.91
4	42.53	41.87	0.61	4.83	5.5	0.0	4.16	5.65	6.8
4*	63.84	61.9	2.52	19.88	9.94	0.0	25.47	25.94	32.31
5	43.06	42.6	0.68	5.38	5.76	0.0	4.69	5.48	7.49
5*	70.45	71.29	2.61	20.54	12.12	0.0	32.08	32.14	38.09
fixed	41.3	41.3	0.0	0.0	6.62	0.0	2.93	3.83	4.53
rolling	40.07	40.49	0.2	1.59	3.5	0.0	1.7	3.45	4.32

Table D.8: Test statistic of one-sided t-test for Experiments 0, 1, 2, 3, 4 and 5 with each other with H_0: MAE of Experiment x is greater or equal the MAE of Experiment y for all validation lots with a lot start date in 2021.

		y											
		0	0*	1	1*	2	3	4	4*	5	5*	fixed	rolling
x	0	0	-2.07	-0.96	0.4	-0.17	1.12	-3.05	-8.88	-2.25	-10.72	-0.12	0.64
	0*	2.07	0	0.86	2.31	1.93	2.92	-0.34	-8.05	-0.12	-9.9	2.08	2.56
	1	0.96	-0.86	0	1.21	0.83	1.71	-1.31	-8.35	-0.98	-10.18	0.92	1.4
	1*	-0.4	-2.31	-1.21	0	-0.55	0.61	-3.3	-8.95	-2.49	-10.78	-0.55	0.18
	2	0.17	-1.93	-0.83	0.55	0	1.28	-2.85	-8.83	-2.11	-10.67	0.07	0.8
	3	-1.12	-2.92	-1.71	-0.61	-1.28	0	-4.3	-9.14	-3.14	-10.97	-1.41	-0.48
	4	3.05	0.34	1.31	3.3	2.85	4.3	0	-8.07	0.22	-9.94	3.17	3.75
	4*	8.88	8.05	8.35	8.95	8.83	9.14	8.07	0	8.02	-1.72	8.89	9.03
	5	2.25	0.12	0.98	2.49	2.11	3.14	-0.22	-8.02	0	-9.88	2.28	2.76
	5*	10.72	9.9	10.18	10.78	10.67	10.97	9.94	1.72	9.88	0	10.73	10.87
	fixed	0.12	-2.08	-0.92	0.55	-0.07	1.41	-3.17	-8.89	-2.28	-10.73	0	0.84
	rolling	-0.64	-2.56	-1.4	-0.18	-0.8	0.48	-3.75	-9.03	-2.76	-10.87	-0.84	0

Table D.9: p-value of one-sided t-test for Experiments 0, 1, 2, 3, 4 and 5 with each other with H_0: MAE of Experiment x is greater or equal the MAE of Experiment y for all validation lots with a lot start date in 2021.

		y											
		0	0*	1	1*	2	3	4	4*	5	5*	fixed	rolling
x	0	0.5	0.02	0.17	0.66	0.43	0.87	0	0	0.01	0	0.45	0.74
	0*	0.98	0.5	0.8	0.99	0.97	1	0.37	0	0.45	0	0.98	0.99
	1	0.83	0.2	0.5	0.89	0.8	0.95	0.1	0	0.16	0	0.82	0.92
	1*	0.34	0.01	0.11	0.5	0.29	0.73	0	0	0.01	0	0.29	0.57
	2	0.57	0.03	0.2	0.71	0.5	0.9	0	0	0.02	0	0.53	0.79
	3	0.13	0	0.05	0.27	0.1	0.5	0	0	0	0	0.08	0.32
	4	1	0.63	0.9	1	1	1	0.5	0	0.59	0	1	1
	4*	1	1	1	1	1	1	1	0.5	1	0.04	1	1
	5	0.99	0.55	0.84	0.99	0.98	1	0.41	0	0.5	0	0.99	1
	5*	1	1	1	1	1	1	1	0.96	1	0.5	1	1
	fixed	0.55	0.02	0.18	0.71	0.47	0.92	0	0	0.01	0	0.5	0.8
	rolling	0.26	0.01	0.08	0.43	0.21	0.68	0	0	0	0	0.2	0.5

Table D.10: Test statistic of one-sided t-test for Experiments 2, 3 and industrial standard approaches with each other with H_0: MAE of Experiment x is greater or equal the MAE of Experiment y for all validation lots (a), all validation lots with a cycle time smaller than 35 days (b) and all validation lots with a cycle time higher than 48 days (c)

		y											
		all				< 35				> 48			
		2	3	fixed	rolling	2	3	fixed	rolling	2	3	fixed	rolling
x	2	0	5.34	2.98	-1.55	0	1.57	-8.5	-3.31	0	5.4	2.97	-0.48
	3	-5.34	0	-3.02	-6.13	-1.57	0	-14.94	-4.26	-5.4	0	-3.4	-6.48
	fixed	-2.98	3.02	0	-4.19	8.5	14.94	0	0.44	-2.97	3.4	0	-3.92
	rolling	1.55	6.13	4.19	0	3.31	4.26	-0.44	0	0.48	6.48	3.92	0

List of Figures

List of Tables

List of Publications

Journal articles

Kai Schelthoff, Christoph Jacobi, Eva Schlosser, David Plohmann, Michel Janus, and Kai Furmans. Feature selection for waiting time predictions in semiconductor wafer fabs. *IEEE Transactions on Semiconductor Manufacturing*, 2022.

Bibliography

Scikit-optimize: Sequential model-based optimization in python, 2020. URL scikit-optimize.github.io/stable/index.html.

Reinheits- und reinraumklassen gemäß din en iso und eg-gmp-leitfaden, 2021. URL www.bc-technology.de/reinraumservice/reinraumqualifizierung/reinraumklassen/.

Elif Akcalt, Kazunori Nemoto, and Reha Uzsoy. Cycle-time improvements for photolithography process in semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 14(1):48–56, 2001. ISSN 08946507. doi: 10.1109/66.909654.

Raha Akhavan-Tabatabaei, Shengwei Ding, and J. George Shanthikumar. A method for cycle time estimation of semiconductor manufacturing toolsets with correlations. In *Proceedings of the 2009 Winter Simulation Conference (WSC)*, 2009. doi: 10.1109/WSC.2009.5429165.

Douglas G Altman and J Martin Bland. Standard deviations and standard errors. *Bmj*, 331(7521):903, 2005.

Dieter Arnold and Kai Furmans. *Materialfluss in Logistiksystemen*. VDI-Buch. Springer, Berlin and Heidelberg, 6., erw. aufl. edition, 2009. ISBN 978-3-642-01404-8.

Semiconductor Industry Association. Wsts semiconductor market forecast fall 2021, 2021. URL https://www.wsts.org/esraCMS/extension/media/f/WST/5263/WSTS_nr-2021_11.pdf.

Phillip Backus, Mani Janakiram, Shahin Mowzoon, George C. Runger, and Amit Bhargava. Factory cycle-time prediction with a data-mining approach. *IEEE Transactions on Semiconductor Manufacturing*, 19(2):252–258, 2006. doi: 10.1109/TSM.2006.873400.

Chaya Bakshi. Random forest regression, 2020. URL `https://levelup.gitconnected.com/random-forest-regression-209c0f354c84`.

Marion Baumann. *Discrete Time Analysis of Multi-Queue Systems with Multiple Departure Streams in Material Handling and Production under Different Service Rules*, volume 94. KIT Scientific Publishing, 2020.

Louis G Birta and Gilbert Arbez. *Modelling and simulation*. Springer, 2013.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

Steven M Brown, Thomas Hanschke, Ingo Meents, Benjamin R Wheeler, and Horst Zisgen. Queueing model improves ibm's semiconductor capacity and lead-time management. *Interfaces*, 40(5):397–407, 2010.

Ondrej Burkacky, Johannes Deichmann, and Jan-Paul Stein. Mapping the automotive software-and-electronics landscape through 2030. *Mckinsey & Company: New York, NY, USA*, 2019.

Birkan Can and Cathal Heavey. A demonstration of machine learning for explicit functions for cycle time prediction using mes data. In *2016 Winter Simulation Conference (WSC)*, pages 2500–2511. IEEE, 2016.

Shiladitya Chakravorty and Nagendra N. Nagarur. An artificial neural network based algorithm for real time dispatching decisions. *ASMC (Advanced Semiconductor Manufacturing Conference) Proceedings*, 2020. doi: 10.1109/ASMC49169.2020.9185213.

Pei-Chann Chang, Yen-Wen Wang, and Ching-Jung Ting. A fuzzy neural network for the flow time estimation in a semiconductor manufacturing factory. *International Journal of Production Research*, 46(4):1017–1029, 2008. doi: 10.1080/00207540600905620.

Tin-Chih Chen and Yu Cheng Lin. A collaborative fuzzy-neural approach for internal due date assignment in a wafer fabrication plant. *International Journal of Innovative Computing, Information and Control*, 7(9):5193–5210, 2011.

Toly Chen. Incorporating fuzzy c-means and a back-propagation network ensemble to job completion time prediction in a semiconductor fabrication factory. *Fuzzy Sets and Systems*, 158(19):2153–2168, 2007. ISSN 01650114. doi: 10.1016/j.fss.2007.04.013.

Toly Chen. Job cycle time estimation in a wafer fabrication factory with a bi-directional classifying fuzzy-neural approach. *The International Journal of Advanced Manufacturing Technology*, 56(9-12):1007–1018, 2011. ISSN 0268-3768. doi: 10.1007/s00170-011-3228-3.

Toly Chen. Estimating job cycle time in a wafer fabrication factory: A novel and effective approach based on post-classification. *Applied Soft Computing Journal*, 40:558–568, 2016. doi: 10.1016/j.asoc.2015.12.017.

Toly Chen and Yi-Chi Wang. Incorporating the fcm–bpn approach with nonlinear programming for internal due date assignment in a wafer fabrication plant. *Robotics and Computer-Integrated Manufacturing*, 26(1):83–91, 2010. ISSN 0736-5845. doi: https://doi.org/10.1016/j.rcim.2009.04.001.

Toly Chen and Yi-Chi Wang. An iterative procedure for optimizing the performance of the fuzzy-neural job cycle time estimation approach in a wafer fabrication factory. *Mathematical Problems in Engineering*, 2013, 2013. doi: https://doi.org/10.1155/2013/740478.

Toly Chen, Yi-Chi Wang, Yu-Cheng Lin, and Kai-Hsiang Yang. Estimating job cycle time in semiconductor manufacturing with an ann approach equally dividing and post-classifying jobs. *Materials Science Forum*, 594:469–474, 2008.

Toly Chen, Yi-Chi Wang, and Horng-Ren Tsai. Lot cycle time prediction in a ramping-up semiconductor manufacturing factory with a som-fbpn-ensemble approach with multiple buckets and partial normalization. *International Journal of Advanced Manufacturing Technology*, 42(11-12):1206–1216, 2009a. doi: 10.1007/s00170-008-1665-4.

Toly Chen, Yi-Chi Wang, and H.-C. Wu. A fuzzy-neural approach for remaining cycle time estimation in a semiconductor manufacturing factory-a simulation study. *International Journal of Innovative Computing, Information and Control*, 5(8):2125–2139, 2009b.

Chen-Fu Chien, Chia-Yu Hsu, and Chih-Wei Hsiao. Manufacturing intelligence to forecast and reduce semiconductor cycle time. *Journal of Intelligent Manufacturing*, 23(6):2281–2294, 2012. ISSN 0956-5515. doi: 10.1007/s10845-011-0572-y.

Shu Hsing Chung and Hung Wen Huang. Cycle time estimation for wafer fab with engineering lots. *IIE Transactions*, 34(2):105–118, 2002. ISSN 0740-817X. doi: 10.1080/07408170208928854.

Raphael Couronné, Philipp Probst, and Anne-Laure Boulesteix. Random forest versus logistic regression: a large-scale benchmark experiment. *BMC bioinformatics*, 19(1):270, 2018.

Louis De Branges. The stone-weierstrass theorem. *Proceedings of the American Mathematical Society*, 10(5):822–824, 1959.

Jijie Deng, Zhengcai Cao, and Min Liu. A bottleneck prediction and rolling horizon scheme combined dynamic scheduling algorithm for semiconductor wafer fabrication. In *Proceedings of the 11th IEEE International Conference on Networking, Sensing and Control*, pages 58–63, 2014. doi: 10.1109/ICNSC.2014.6819600.

Kean Dequeant, Philippe Vialletelle, Pierre Lemaire, and Marie-Laure Espinouse. A literature review on variability in semiconductor manufacturing: The next forward leap to industry 4.0. pages 2598–2609, 2016. doi: 10.1109/WSC.2016.7822298.

Maximilian Dilefeld, Sebastian Rank, and Thorsten Schmidt. Enhancing prediction quality of fab simulation by advanced cycle time modelling. In Sophia Keil, Rainer Lasch, Fabian Lindner, and Jacob Lohmer, editors, *Digital Transformation in Semiconductor Manufacturing*, pages 35–42, 2020.

Norman R Draper and Harry Smith. *Applied regression analysis*, volume 326. John Wiley & Sons, 1998.

Aaron Fisher, Cynthia Rudin, and Francesca Dominici. Model class reliance: Variable importance measures for any machine learning model class, from the "rashomon" perspective. *arXiv preprint arXiv:1801.01489*, 68, 2018.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of online learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer, 1996.

Kai Furmans. *Bedientheoretische Methoden als Hilfsmittel der Materialflussplanung*. dissertation, Karlsruhe Institute of Technology, 2000. URL `https://core.ac.uk/download/pdf/197517191.pdf`.

Kai Furmans. A framework of stochastic finite elements for models of material handling systems. In R. Meller, editor, *Progress in material handling research*. The Material Handling Institute, Charlotte (NC), 2004. ISBN 1-88278-011-6.

Kai Furmans, George Liberopoulos, Marion Rimmele, and Olaf Zimmermann. Review of models for large scale manufacturing networks. In *Proceedings 11th conference on stochastic modeling of manufacturing and service operations (SMMSO 2017)*, pages 251–260, 2017.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning: Das umfassende handbuch : grundlagen, aktuelle verfahren und algorithmen, neue forschungsansätze*. Verlags GmbH & Co. KG, Frechen, 1. auflage edition, 2018. ISBN 978-3-95845-701-0.

Winfried K Grassmann and Joti L Jain. Numerical solutions of the waiting time distribution and idle time distribution of the arithmetic gi/g/1 queue. *Operations Research*, 37(1):141–150, 1989.

Uma G. Gupta. Using citation analysis to explore the intellectual base, knowledge dissemination, and research impact of interfaces (1970–1992). *Interfaces*, 27 (2):85–101, 1997.

Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

Michael Hassoun. On improving the predictability of cycle time in an nvm fab by correct segmentation of the process. *IEEE Transactions on Semiconductor Manufacturing*, 26(4):613–618, 2013. doi: 10.1109/TSM.2013.2283262.

Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

Tin Kam Ho. A data complexity analysis of comparative advantages of decision forest constructors. *Pattern Analysis & Applications*, 5(2):102–112, 2002.

Andreas Hocker, Peter Speckmayer, Jorg Stelzer, Jan Therhaag, Eckhard von Toerne, Helge Voss, Moritz Backes, Carli Tancredi, Or Cohen, Asen Christov, Dominik Dannheim, et al. Tmva-toolkit for multivariate data analysis with root: Users guide. Technical report, 2007.

John L Hodges. The significance probability of the smirnov two-sample test. *Arkiv för Matematik*, 3(5):469–486, 1958.

Liam Y Hsieh, Kuo-Hao Chang, and Chen-Fu Chien. Efficient development of cycle time response surfaces using progressive simulation metamodeling. *International Journal of Production Research*, 52(10):3097–3109, 2014.

Nantian Huang, Guobo Lu, and Dianguo Xu. A permutation importance-based feature selection method for short-term electricity load forecasting using random forest. *Energies*, 9(10):767, 2016.

Michel Janus. Search for supersymmetry in final states with jets, missing transverse momentum and at least one τ lepton with the atlas experiment. 2013.

Rachel T. Johnson, John W. Fowler, and Gerald T. Mackulak. A discrete event simulation model simplification technique. In *Proceedings of the Winter Simulation Conference, 2005.*, 2005. doi: 10.1109/WSC.2005.1574503.

Adar A. Kalir. Segregating preventive maintenance work for cycle time optimization. *IEEE Transactions on Semiconductor Manufacturing*, 26(1):125–131, 2013. ISSN 08946507. doi: 10.1109/TSM.2012.2234152.

Andreas Klemmt. *Ablaufplanung in der Halbleiter-und Elektronikproduktion: hybride Optimierungsverfahren und Dekompositionstechniken*. Springer-Verlag, 2012.

Michael S. Lane, Ali H. Mansour, and John L. Harpell. Operations research techniques: A longitudinal update 1973–1988. *Interfaces*, 23(2):63–68, 1993.

Averill M. Law and W. David Kelton. *Simulation modeling and analysis*, volume 3. McGraw-Hill New York, 2000.

Howard Levene. Robust tests for equality of variances in contribution to probability and statistics,(ed) 1. *Olkin: Stanford University Press, Palo Alto*, 1960.

Yu-Hsin Lin and Ching-En Lee. A total standard wip estimation method for wafer fabrication. *European Journal of Operational Research*, 131(1):78–94, 2001. ISSN 0377-2217. doi: 10.1016/S0377-2217(99)00446-4.

Lukas Lingitz, Viola Gallina, Fazel Ansari, Dávid Gyulai, András Pfeiffer, Wilfried Sihn, and László Monostori. Lead time prediction using machine learning algorithms: A case study by a semiconductor manufacturer. *Procedia CIRP*, 72:1051–1056, 2018. ISSN 22128271. doi: 10.1016/j.procir.2018.03.148.

Stephen Marsland. *Machine learning: an algorithmic perspective*. CRC press, 2015.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Yair Meidan, Boaz Lerner, Gad Rabinowitz, and Michael Hassoun. Cycle-time key factor identification and prediction in semiconductor manufacturing using machine learning and data mining. *IEEE Transactions on Semiconductor Manufacturing*, 24(2):237–248, 2011. ISSN 08946507. doi: 10.1109/TSM.2011.2118775.

Lars Mönch, John W. Fowler, Stéphane Dauzère-Pérès, Scott J. Mason, and Oliver Rose. A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. *Journal of Scheduling*, 14(6):583–599, 2011. ISSN 1094-6136. doi: 10.1007/s10951-010-0222-9.

James R. Morrison and Donald P. Martin. Practical extensions to cycle time approximations for the $g/g/m$-queue with applications. *IEEE Transactions on Automation Science and Engineering*, 4(4):523–532, 2007. doi: 10.1109/TASE.2007.905975.

W. L. Pearn, Yu-Ting Tai, and J. H. Lee. Statistical approach for cycle time estimation in semiconductor packaging factories. *IEEE Transactions on electronics packaging manufacturing*, 32(3):198–205, 2009.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Èdouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Markus Pfeffer. Simulationsgestützte untersuchung von logistischen optimierungsstrategien bei halbleiterfertigungsprozessen. 2012.

Xueheng Qiu, Le Zhang, Ye Ren, Ponnuthurai N. Suganthan, and Gehan Amaratunga. Ensemble deep learning for regression and time series forecasting.

In *2014 IEEE symposium on computational intelligence in ensemble learning (CIEL)*, pages 1–6. IEEE, 2014.

Stewart Robinson. *Simulation: the practice of model development and use*. Palgrave Macmillan, 2014.

Juan José Rodriguez, Ludmila I Kuncheva, and Carlos J Alonso. Rotation forest: A new classifier ensemble method. *IEEE transactions on pattern analysis and machine intelligence*, 28(10):1619–1630, 2006.

Roland E. A. Schelasin. Using static capacity modeling and queuing theory equations to predict factory cycle time performance in semiconductor manufacturing. In *Proceedings of the 2011 Winter Simulation Conference (WSC)*, pages 2040–2049, 2011. doi: 10.1109/WSC.2011.6147917.

Roland E. A. Schelasin. Estimating wafer processing cycle time using an improved g/g/m queue. In *2013 Winter Simulations Conference (WSC)*, pages 3789–3795. IEEE, 2013.

Marc Schleyer and Kai Furmans. An analytical method for the calculation of the waiting time distribution of a discrete time g/g/1-queueing system with batch arrivals. *OR Spectrum*, 29(4):745–763, 2007.

Wolfgang Scholl and Joerg Domaschke. Implementation of modeling and simulation in semiconductor wafer fabrication with time constraints between wet etch and furnace operations. *IEEE Transactions on Semiconductor Manufacturing*, 13(3):273–277, 2000.

Georg Seidel, Patrick Preuss, Cevahir Canbolat, Soo Leen Low, Chew Wye Chan, Boon Ping Gan, Ching Foong Lee, Prakash Manogaran, and Aik Ying Tang. An integration of static and dynamic capacity planning for a ramping fab. In *2019 Winter Simulation Conference (WSC)*, pages 2304–2311, 2019. doi: 10.1109/WSC40007.2019.9004674.

J. George Shanthikumar, Shengwei Ding, and Mike Tao Zhang. Queueing theory for semiconductor manufacturing systems: A survey and open problems. *IEEE*

Transactions on Automation Science and Engineering, 4(4):513–522, 2007. doi: 10.1109/TASE.2007.906348.

Appa Iyer Sivakumar and Chin Soon Chong. A simulation based analysis of cycle time distribution, and throughput in semiconductor backend manufacturing. *Computers in Industry*, 45(1):59–78, 2001.

Student. The probable error of a mean. *Biometrika*, pages 1–25, 1908.

S. Suresh and W. Whitt. Arranging queues in series: A simulation experiment. *Management Science*, 36(9):1080–1091, 1990.

Yu-Ting Tai, W. L. Pearn, and J. H. Lee. Cycle time estimation for semiconductor final testing processes with weibull-distributed waiting time. *International Journal of Production Research*, 50(2):581–592, 2012. doi: 10.1080/00207543.2010.543938.

Israel Tirkel. Cycle time prediction in wafer fabrication line by applying data mining methods. In *2011 IEEE/SEMI Advanced Semiconductor Manufacturing Conference*, pages 1–5, 2011. doi: 10.1109/ASMC.2011.5898218.

Israel Tirkel. Forecasting flow time in semiconductor manufacturing using knowledge discovery in databases. *International Journal of Production Research*, 51 (18):5536–5548, 2013. doi: 10.1080/00207543.2013.787168.

Reha Uzsoy, Chung Yee Lee, and Louis A. Martin-Vega. A review of production planning and scheduling models in the semiconductor industry part 1: System characteristics, performcance evaluation and production planning. *IIE Transactions*, 24(4):47–60, 1992. ISSN 0740-817X. doi: 10.1080/07408179208964233.

CPL Veeger, LFP Etman, E Lefeber, IJBF Adan, J Van Herk, and JE Rooda. Predicting cycle time distributions for integrated processing workstations: an aggregate modeling approach. *IEEE Transactions on Semiconductor Manufacturing*, 24(2):223–236, 2010a.

CPL Veeger, LFP Etman, Joost van Herk, and Jacobus E Rooda. Generating ct-th-pm surfaces using ept-based aggregate modelling. *Journal of Simulation*, 4(4):242–254, 2010b.

John Carl Villanueva. How many atoms are there in the universe?, 2009. URL `www.universetoday.com/36302/atoms-in-the-universe/`.

Karl-Heinz Waldmann and Werner E Helm. *Simulation stochastischer Systeme*. Springer, 2016.

Junliang Wang and Jie Zhang. Big data analytics for forecasting cycle time in semiconductor wafer fabrication system. *International Journal of Production Research*, 54(23):7231–7244, 2016a. doi: 10.1080/00207543.2016.1174789.

Junliang Wang and Jie Zhang. Big data analytics for forecasting cycle time in semiconductor wafer fabrication system. *International Journal of Production Research*, 54(23):7231–7244, 2016b. doi: 10.1080/00207543.2016.1174789.

Junliang Wang and Jie Zhang. Big data analytics for forecasting cycle time in semiconductor wafer fabrication system. *International Journal of Production Research*, 54(23):7231–7244, 2016c.

Junliang Wang, Jungang Yang, Jie Zhang, Xiaoxi Wang, and Wenjun (Chris) Zhang. Big data driven cycle time parallel prediction for production planning in wafer manufacturing. *Enterprise Information Systems*, 12(6):714–732, 2018a. doi: 10.1080/17517575.2018.1450998.

Junliang Wang, Jie Zhang, and Xiaoxi Wang. A data driven cycle time prediction with feature selection in a semiconductor wafer fabrication system. *IEEE Transactions on Semiconductor Manufacturing*, 31(1):173–182, 2018b. ISSN 08946507. doi: 10.1109/TSM.2017.2788501.

Junliang Wang, Peng Zheng, and Jie Zhang. Big data analytics for cycle time related feature selection in the semiconductor wafer fabrication system. *Computers & Industrial Engineering*, 143, 2020. doi: 10.1016/j.cie.2020.106362.

Yu-Cheng Wang, Horng-Ren Tsai, and Toly Chen. A selectively fuzzified back propagation network approach for precisely estimating the cycle time range in wafer fabrication. *Mathematics*, 9(12):1430, 2021.

Bernard L Welch. The generalization of 'student's' problem when several different population varlances are involved. *Biometrika*, 34(1-2):28–35, 1947.

Wikipedia. Student´s t-distribution, 2010. URL `https://en.wikipedia.org/wiki/File:Student_t_pdf.svg`.

Feng Yang, Bruce E. Ankenman, and Barry L. Nelson. Estimating cycle time percentile curves for manufacturing systems via simulation. *INFORMS Journal On Coputing*, 20(4):499–666, 2008. doi: 10.1287/ijoc.1080.0272.

Feng Yang, Jingang Liu, Barry L Nelson, Bruce E Ankenman, and Mustafa Tongarlak. Metamodelling for cycle time-throughput-product mix surfaces using progressive model fitting. *Production Planning and Control*, 22(1):50–68, 2011.

Peng Zhang, Xinming Zhao, Xia Sheng, and Jie Zhang. An imperialist competitive algorithm incorporating remaining cycle time prediction for photolithography machines scheduling. *IEEE Access*, 6:66787–66797, 2018.

Horst Zisgen, Ingo Meents, Benjamin R. Wheeler, and Thomas Hanschke. A queueing network based system to model capacity and cycle time for semiconductor fabrication. In *2008 Winter Simulation Conference*, 2008. doi: 10.1109/WSC.2008.4736303.