Predicting flow time distributions in workstations with dispatching: an aggregate modeling approach

C.P.L. Veeger, L.F.P. Etman, A.A.J. Lefeber, I.J.B.F. Adan and J.E. Rooda
Eindhoven University of Technology, {c.p.l.veeger, l.f.p.etman, i.j.b.f.adan, a.a.j.lefeber, j.e.rooda}@tue.nl

For manufacturing workstations, there is a trade-off between throughput and meeting product due dates. Models for predicting flow time distributions are helpful in this respect. A model of a workstation typically has to include the process time distributions of the machines in the workstation, the dispatching rule, and other disturbances that affect the flow time performance. A detailed simulation model may be used to predict the flow time distributions of a workstation. However, such a model typically involves considerable development time, and obtaining all model parameters may be difficult. In this paper we propose a single-server aggregate model with a workload-dependent aggregate process time distribution, and a workload-dependent overtaking distribution. The model involves little development time. The process time and overtaking distribution in the aggregate model are determined from lot arrival and departure events measured at the considered workstation. Two simulation test cases are included to demonstrate the proposed method.

Key words: Flow time distributions, simulation, aggregate modeling, factory dynamics

1. Introduction

Predicting flow time distributions as a function of throughput is helpful in production planning of manufacturing workstations. We assume here that a workstation consists of a number of parallel machines which are feeded by an infinite buffer. The way lots are selected from the buffer to be processed on the machines is referred to as the dispatching rule. With flow time, we mean the sum of queue time and process time of a lot at a specific workstation. The throughput is the number of lots processed per time unit. From a flow time distribution, quantiles can be derived. For instance, the 95% quantile defines the flow time in which 95% of the lots are completed.

Only few analytical models to predict flow time distributions of workstations exist. Queueing systems for which analytical models are available are e.g. the $M/M/1$ queue (Cohen 1982), and the $M/M/m$ queue (Adan and Resing 2002). Simulation is often the only option to calculate flow time distributions. For example, Sivakumar and Chong (2001) used a simulation-based analysis of flow time distributions in semiconductor back-end manufacturing. Simulation-based analysis is computationally expensive. Yang et al. (2008) therefore proposed to derive a metamodel from a detailed simulation model, which they use to derive flow time quantiles as a function of the throughput. The development of a detailed simulation model is usually time-consuming effort and it may be difficult to obtain all model parameters.

Rose (2000) investigated the use of a simplified simulation model, in which the bottleneck workstation is modeled in detail and the remaining workstations in the network are lumped in a delay distribution. He concluded that the proposed model inaccurately estimates flow time distributions for certain scenarios. Following up on this work, Rose (2007) introduced a utilization-dependent delay distribution determined by running a full detail simulation model at various utilization levels.

In this paper, we propose an alternative simplified simulation model to predict flow time distributions for multi-machine workstations with dispatching. The proposed model is a single-server aggregate queueing model. The lumped parameters of the aggregate model can be determined directly from measured arrival and departure events at the workstation at a single utilization

level. Our starting point is the Effective Process Time (EPT)-based aggregate modeling method presented in Kock et al. (2008).

The EPT was originally defined by Hopp and Spearman (2000) as 'the time seen by a lot at a workstation from a logistical point of view'. The mean and variance of the EPT may be calculated from the raw process time and the various outage delay distributions in the process, and used in analytical equations representing the $G/G/m$ system (Hopp and Spearman 2000). Data of the various distributions may not always be available. Jacobs et al. (2003) therefore derived the EPT distribution parameters directly from arrivals and departures of lots at the workstation. Kock et al. (2008) proposed a G/G/m alike aggregate model with workload-dependent EPT distributions, motivated by integrated processing types of machines which may have multiple lots in process at the same time. Both Jacobs et al. (2003) and Kock et al. (2008) developed EPT-based aggregate models for mean flow time performance analysis. Due to the First-Come-First-Serve (FCFS) assumption in their aggregate model flow time distributions are not accurately predicted.

The single server aggregate model proposed in this paper also has a workload-dependent EPT distribution, but additionally includes a probability distribution for lot overtaking. The overtaking distribution depends on the queue length. The workload-dependent EPT distribution and overtaking distribution are measured from arrival and departure events. Two simulation test cases illustrate the proposed method: an $M/M/m$ workstation with FCFS, Last-Come-First-Serve (LCFS) and random dispatching, and a workstation with three integrated processing machines with qualification-based dispatching. The simulation results show that the new method is able to accurately predict the flow time distribution as a function of the throughput.

The the paper is outlined as follows: the EPT-based aggregate modeling method of Kock et al. (2008) is summarized in Section 2. The new method is explained in Section 3. Next, we illustrate the method by two simulation test cases in Section 4. Finally, we present conclusions in Section 5.

2. EPT-based aggregate modeling of multi-processing workstations

Our starting point is the EPT-based aggregate modeling method of Kock et al. (2008). Kock et al. (2008) approximate a multi-processing workstation by a multi-server station similar to a $G/G/m$ approximation, with the difference that the mean and variance of the process time distribution depend on the number of lots in the system. Although Kock et al. (2008) allow multiple parallel servers in the aggregate model we consider here the single server aggregation case.

Figure 1a shows an example of a multi-processing workstation. The example workstation consists of four machines. Each machine is represented by three process steps with a one-place buffer in between. In the aggregate model, lots arrive according to the same arrival process as in the real workstation. The single server aggregate model according to Kock et al. (2008) consists of an infinite, FCFS buffer that feeds the aggregate server. After completion of service, lots leave the system (no blocking after service). In the aggregate model service starts if the server is, or becomes, idle and lots are present in the queue (non-idling assumption). The process time of a lot is sampled from a distribution at the moment it starts processing according to the aggregate model. Kock et al. (2008) use a gamma distribution, in which the mean and the variance depend on the momentary number of lots in the system.

Let w be the number of lots in the system at the process start of lot i (including lot i itself). For each work-in-progress (wip) level a so-called bucket is defined. If there are w lots in the system upon process start of lot i, then the process time of lot i is sampled from a gamma distribution with distribution parameters corresponding to bucket w. Kock et al. (2008) use an independent process time distribution for each bucket. A highest bucket N is defined. Bucket N contains all process times registered with N or more lots in the system.

The input required for this model consists of one EPT-distribution per bucket, hence N distributions are required. To determine these EPT-distributions, arrival and departure data is used.

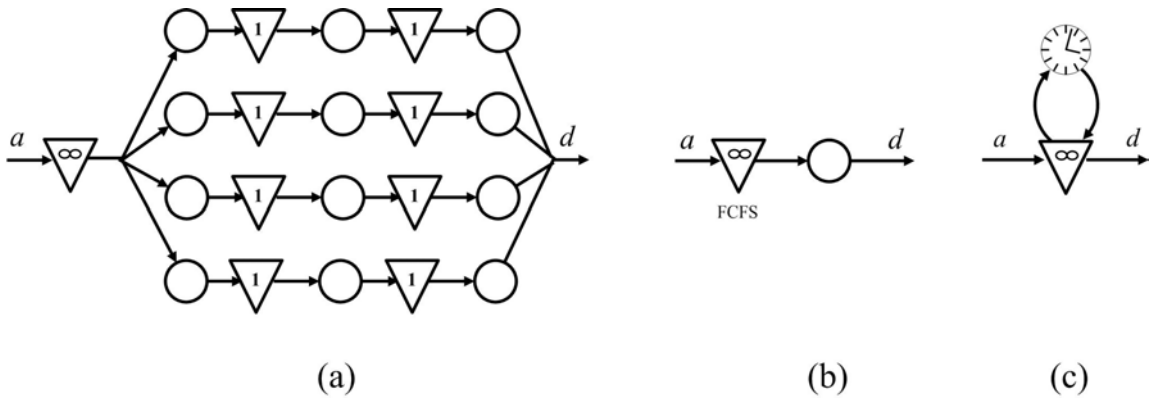

Figure 1 **An example of a workstation (a), $m = 1$ aggregate model according to Kock et al. (2008) (b), the aggregate model proposed in this paper (c)**

For each lot i departing from the considered workstation, departure time d_i is collected, as well as corresponding arrival time a_i of the lot in the buffer of the station. This arrival and departure data is translated into EPT-realizations using an EPT algorithm. From the EPT-realizations, EPT distributions per bucket are calculated.

3. Aggregate modeling method with lot overtaking

We propose a new infinitely buffered single server aggregate queueing model with a workload-dependent process time distribution and a workload dependent overtaking distribution. Overtaking may arise due to parallel processing, or due to dispatching policies other than FCFS. [1] We assume that dispatching does not use information about the lot itself (e.g. Shortest Processing Time).

3.1. The aggregate model

Figure 1c visualizes the proposed aggregate model. The queue contains all w lots that are currently in the system. During service, lots stay in the queue (unlike common queue-server models). If the service time has elapsed, the lot that is currently first in the queue leaves the system. Upon arrival of a new lot i, it is determined how many lots in the queue are overtaken by lot i. The number of lots to overtake $k_i \in \{0, 1, ..., w\}$ is sampled from a probability distribution that depends on the amount of lots already in the system when lot i arrives (not including lot i). The arriving lot i is placed on position $w - k_i$ in the queue. For example, if $w = 1$ upon arrival of lot i, there is a probability that no lots are overtaken ($k_i = 0$), and a probability that one lot is overtaken ($k_i = 1$). If no lots are overtaken, lot i is placed at the end of the queue (position $1 - 0 = 1$). If one lot is overtaken, lot i is placed ahead of the queue (position $1 - 1 = 0$).

Note that in the aggregate model, the server is not a true physical server: a timer determines when the next lot leaves the queue. The timer starts when: i) a lot arrives while no lots are present in the buffer, or ii) a lot departs while leaving one or more lots behind. When the timer starts, a time period is sampled from a probability distribution that depends on number of lots in the system w upon the timer start. The sampled time period is referred to as an Effective Process Time (EPT). When the EPT is finished, the first lot in the queue (position 0) leaves the system. While the EPT timer is not yet finished, new arriving lots may still overtake *all* lots in the system, including the first lot in the queue.

The input to the proposed aggregate model consists of an EPT distribution per bucket (wip-level) and an overtaking probability function. We assume that the EPT-distributions are gamma

[1] Note that an $m > 1$ aggregation in the model of Kock et al. (2008) can model overtaking due to parallel processing, but not overtaking due to dispatching.

with mean t_e and coefficient of variability c_e. We also assume that the distributions for the various buckets are independent. We denote the overtaking probability function by $P(w,k)$, which is defined as the probability that k lots are overtaken for w lots in the system upon arrival.

3.2. Calculating model parameters

To determine the EPT distributions and the overtaking probability function $P(w,k)$, arrival and departure data is measured from the workstation under consideration. For each lot i departing from the workstation, departure time d_i is collected, as well as the corresponding arrival time a_i of the lot in the buffer of the workstation. From the arrival and departure data, the number of lots overtaken by each lot i as well as the EPT realizations, are calculated using the algorithm shown Appendix A. The algorithm input consists of a lists of events, each event consisting of time τ, event type ev, and lot arrival number i. The event type can be an arrival or a departure of a lot. Lot i is the i^{th} arriving lot at the workstation. The events are sorted in increasing time order.

The EPT algorithm takes the aggregate model viewpoint. Upon an arrival event, a new EPT is started if the lot arrives in an empty system (start time s becomes τ); the corresponding wip-level is stored (variable sw in Figure 6). For every arriving lot, the lot arrival number i and the number of lots in the system upon arrival (aw in Figure 6) are stored. If a departure event occurs, an EPT ends, the EPT being the current time (τ) minus the EPT start time (s). The EPT is written to output along with the number of lots in the system upon the EPT start (sw). Next, the algorithm reconstructs how many lots (k) were overtaken by the departing lot i. This number is equal to the number of lots in the system upon departure of lot i that have a lower arrival number than lot i. The number of overtaken lots (k) and the number of lots in the system upon arrival of lot i (aw) are written. If there are still lots in the system after the departure of lot i, a new EPT start time is stored (s); the corresponding number of lots currently in the system is stored (sw).

The EPT-realizations calculated by the algorithm are assigned to buckets corresponding to the number of lots in the system upon the EPT start (taking into account maximum bucket N). For each bucket, the mean and coefficient of variation of the measured EPT distribution are determined, which are used in the process time gamma distributions of the aggregate model for the respective wip-levels. To obtain $P(w,k)$, overtaking realizations are assigned to buckets corresponding to the number of lots in the system upon arrival.

4. Simulation test cases

We test the new aggregate modeling method on two simulation test cases. Simulation results were generated using the language χ (Hofkamp and Rooda 2007).

4.1. Description of test cases

The test cases are shown in Figure 2. The first test case (Figure 2a) is an $M/M/m$ system with FCFS, LCFS, and random dispatching for which we consider $m = 1$ and $m = 12$. Below, we present results for the $M/M/1$ system with LCFS dispatching, and the $M/M/12$ system with FCFS dispatching. The results of the other variants give similar insights. The second test case (2b) consists of three parallel machines that each consist of two sequential processes. Lots arrive at the workstation according to a Poisson process. Two lot types (being A and B) arrive at the infinite buffer that feeds the parallel machines: 50% is of type A, and 50% is of type B. The first two machines can process both lot types, whereas the third machine can only process recipe A lots. The process time distribution of all process steps is gamma, with coefficient of variability 0.5. The mean process time of the first process step of each machine is 1.0 for both lot types. The mean process time of the second process step of each machine is 2.0 for type A lots, and 4.0 for type B lots. Lots are processed in FCFS order taking into account the machine qualification. If more than one qualified machine is available for processing, the lot is sent to the machine that has been idle the longest (fairness).

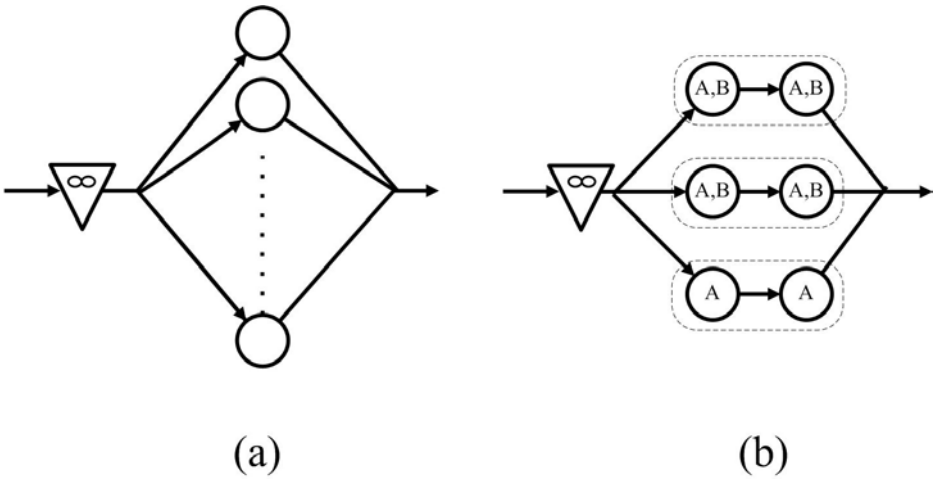

Figure 2 $M/M/m$ **system with FCFS/LCFS/random dispatching (a), and a multi-machine multi-processing workstation with qualification dependent dispatching (b)**

4.2. Calculating model parameters

For each test case, arrivals and departures of 10^6 simulated lots were obtained at a throughput ratio $\delta/\delta_{\max}$ of 0.8. We set maximum EPT bucket N to 15. The algorithm in Appendix A was used to calculate EPT realizations and overtaking realizations k, which were assigned to buckets as explained in the previous section.

Figure 3 plots the mean EPT t_e (left hand side) and the coefficient of variability of the EPT c_e (right hand side) as a function of w. The solid black line represents the $M/M/1$ with LCFS dispatching test case: t_e and c_e are constant. The dashed line represents the $M/M/12$ with FCFS test case: t_e decreases for increasing w, approaching 1/12. This is because for increasing w the inter departure time in the $M/M/12$ system decreases up to $w = 12$. c_e approaches 1 for increasing w, because the inter departure time is exponentially distributed when all (exponential) servers are processing. Finally, the integrated processing workstation test case is represented by the grey lines. The explanation of the behavior is similar to the $M/M/12$ test case.

Figure 4 shows the cumulative overtaking probabilities $\sum_{j=0}^{k} P(w, j)$ as a function of k for several values of w. For the LCFS-$M/M/1$ test case, every arriving lot overtakes all lots in the system except the one in process. Hence, $\sum_{j=0}^{k} P(w, j)$ jumps from 0 to 1 for $k = w - 1$ (recall that k is the number of lots overtaken). For the FCFS-$M/M/12$ test case overtaking will only occur due to the parallel servers. Hence, the maximum number of lots that can be overtaken is 11. Because process times are exponentially distributed, there is an equal probability to overtake $k = 0, .., \min(w, 11)$ lots already in the system. Finally, for the multi-processing workstation test case, Figure 4 shows that for $w \geq 10$ the probability of overtaking 0 or 1 lots is about 50%. This is caused by type B lots that have a relatively high process time and do not overtake many other lots. Figure 4 also shows that there is a probability of about 50% that more than one lot is overtaken. This is caused by type A lots that have a relatively low process time and overtake type B lots.

4.3. Flow time predictions

Figure 5 depicts cumulative flow time distributions for the single server LCFS, the 12-server FCFS, and the multi-processing case for a throughput ratio of 0.6, 0.8, and 0.9. Recall that the aggregate model parameters were obtained for $\delta/\delta_{\max} = 0.8$. The x-axis denotes the flow time φ, the y-axis the cumulative probability $P(X \leq \varphi)$ that the flow time is less than or equal to φ. Flow times are obtained using simulation for 10^6 lots. The solid black lines represent the cumulative flow time distributions of the test case system. The simulated flow time distribution of the FCFS $M/M/12$

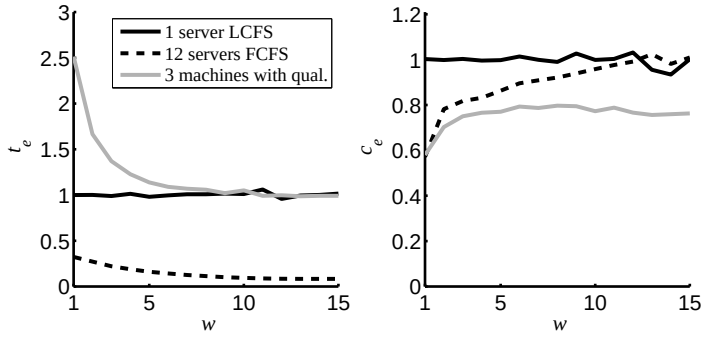

Figure 3 **Mean EPT t_e and coefficient of variability c_e as a function of w for test case 2 and 3**

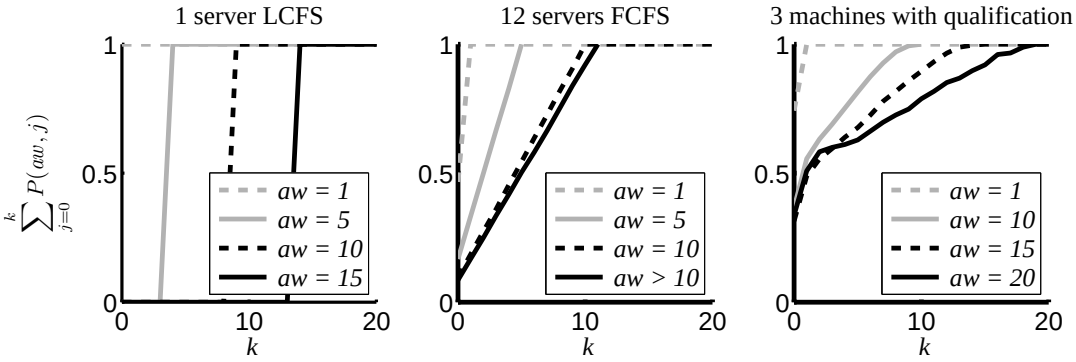

Figure 4 **Cumulative overtaking probabilities as a function of w for test case 2 and 3**

system is verified analytically (Adan and Resing 2002) (not shown in the figure). The grey lines represent the flow time distributions predicted by the method of Kock et al. (2008). The dashed lines give the flow time distributions predicted by the new method.

Figure 5 shows that for the single server LCFS, and the 12-server FCFS system, the new method accurately predicts the flow time distribution, whereas the method of Kock et al. (2008) is significantly less accurate. This is because overtaking is taken into account in the new method, whereas Kock et al. (2008) assume FCFS in their aggregate model. Similar results have been obtained for the single server FCFS system, the single server system with random dispatching, the 12-server LCFS system, and the 12-server system with random dispatching. For the multi-processing case, the improvement is still present, but the observed differences are smaller.

5. Conclusion

In this paper, a novel infinitely buffered single-server aggregate modeling method is proposed to predict flow time distributions for manufacturing workstations with dispatching, assuming dispatching rules that do not use information about the lot itself (e.g. Shortest Processing Time). In the proposed aggregate model, the process time is sampled from a gamma distribution that depends on the momentary workload. Lots entering the infinite buffer have a probability to overtake other lots according to a workload-dependent overtaking distribution. The process time and overtaking distribution are determined from measured arrival and departure events at the workstation.

The proposed method was illustrated by means of two test cases, being an $M/M/m$ system with FCFS, LCFS, and random dispatching, and a workstation with three parallel, qualified machines with two sequential process steps. We found that flow time distributions are accurately determined using the new method and significantly more accurate than those obtained using the method proposed by Kock et al. (2008).

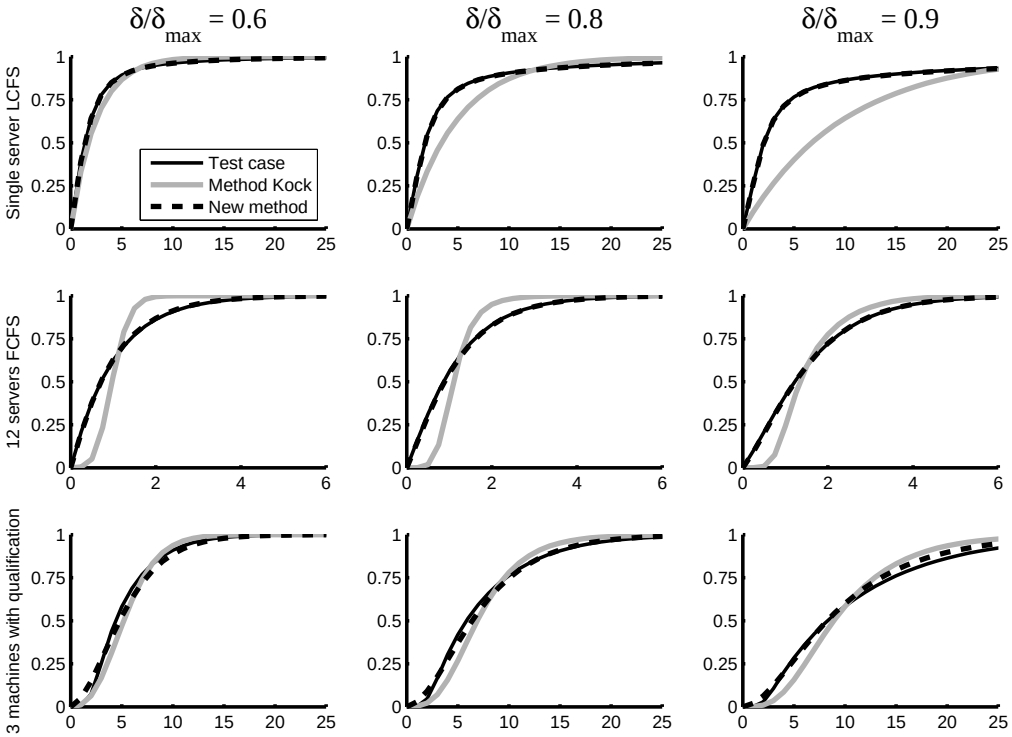

Figure 5 Cumulative flow time distribution of the test case system, and predicted by the method of Kock et al. (2008) and the new method. The x-axis denotes the flow time, the y-axis denotes the cumulative probability.

This paper focussed on a simulation-based analysis of the proposed aggregate modeling method to show its potential. In future research we aim to test the method on data obtained from real manufacturing workstations, and for the aggregation of manufacturing networks. The proposed aggregate model seems also sufficiently simple to provide opportunities for the development of an analytical model representation.

References

Adan, I. J. B. F., J. A. C. Resing. 2002. *Queueing theory*. Department of Mathematics and Computing Science, Eindhoven University of Technology. Lecture notes.

Cohen, J. W. 1982. *The single server queue*. 2nd ed. North-Holland Publication Company, Amsterdam.

Hofkamp, A.T., J.E. Rooda. 2007. *χ 1.0 Reference Manual*. Systems Engineering Group, Eindhoven University of Technology. `http://se.wtb.tue.nl/sewiki/chi/`.

Hopp, W. J., M. L. Spearman. 2000. *Factory Physics: Foundations of Manufacturing Management*. 2nd ed. IRWIN/McGraw-Hill, New York.

Jacobs, J. H., L. F. P. Etman, E. J. J. van Campen, J. E. Rooda. 2003. Characterization of operational time variability using effective process times. *IEEE Transactions on semiconductor manufacturing* **16**(3) 511–520.

Kock, A. A. A., L. F. P. Etman, J. E. Rooda, I. J. B. F. Adan, M. v. Vuuren, A. Wierman. 2008. Aggregate modeling of multi-processing workstations. Eurandom report, `http://www.eurandom.tue.nl/reports/2008`.

Rose, O. 2000. Why do simple wafer fab models fail in certain scenarios? *Proceedings of the 2000 Winter Simulation Conference*. 1481–1490.

Rose, O. 2007. Improved simple simulation models for semiconductor wafer factories. *Proceedings of the 2007 Winter Simulation Conference*. 1708–1712.

Sivakumar, A. I., C. S. Chong. 2001. A simulation based analysis of cycle time distribution, and throughput in semiconductor backend manufacturing. *Computers in Industry* **45** 59–78.

Yang, F., B. E. Ankenman, B. L. Nelson. 2008. Estimating cycle time percentile curves for manufacturing systems via simulation. *INFORMS Journal on Computing* **20**(4) 628–643.

Appendix A: Algorithm

The algorithm used to calculate EPT-realizations and overtaking realizations is depicted in Figure 6. The following variables are used: variable τ denotes the event time, variable ev the event type (arrival **a** or departure **d**), and i the lot arrival number. Furthermore, variable xs is a list that contains for each lot in the system its arrival number, i, and the number of lots in the system upon its arrival, aw. Variable s is used to store the EPT start time. Variable sw denotes the number of lots in the system upon the EPT start. Variable k denotes the number of lots that a lot has overtaken. Function detOvert uses the following additional variables: ys is a list that stores part of list xs. Variable j stores a lot arrival number.

The input of function detOvert consists of list xs and the arrival number i of the departing lot. The function iteratively removes each lot from xs and assigns its arrival number and the number of lots upon its arrival to variables j and aw respectively. If the arrival number of the observed lot is lower than the arrival number i of the departed lot, then (j, as) is concatenated to ys. If the arrival number j of the observed lot is equal to i, the function returns list $ys +\!\!+ xs$, which does not include lot i. Furthermore, the length of ys, and aw are returned. Note that the length of ys is equal to the number of lots that arrived earlier than lot i, but that are still in the system upon the departure of lot i. In other words, the length of ys is equal to the number of lots overtaken by lot i.

```
loop
    read τ, ev, i
    if ev = a :
        if len(xs) = 0 :
            (s, sw) := (τ, 1)
        end if
        xs := xs ++ [(i, len(xs))]
    elseif ev = d :
        write τ − s, sw
        (xs, k, w) := detOvert(xs, i)
        write k, w
        if len(xs) > 0 :
            (s, sw) := (τ, len(xs))
        end if
    end if
end loop
```

```
function detOvert(xs, i)
    ys := []
    while len(xs) > 0 :
        (j, w) := head(xs); xs := tail(xs)
        if j < i :
            ys := ys ++ [(j, w)]
        elseif j = i :
            return (ys ++ xs, len(ys), w)
        end if
    end while
```

Figure 6 EPT Algorithm (left) and function detOvert (right)