Advances in Simulation for
Production and Logistics Applications
Markus Rabe (ed.)
Stuttgart, Fraunhofer IRB Verlag 2008

Generating Cycle time-Throughput-Product Mix Surfaces Using EPT-based Aggregate Modeling

Generierung von Durchlaufzeit-Durchsatz-Produktmix-Oberflächen mit EPT-basierter aggregierter Modellierung

Casper P.L. Veeger, L.F.P. Etman, and J.E. Rooda
Systems Engineering Group, Department of Mechanical Engineering
Eindhoven University of Technology, the Netherlands

Abstract: Cycle time-throughput-product mix (CT-TH-PM) surfaces of bottleneck workstations are helpful to make a trade-off between throughput, mean cycle time, and the product mix. To generate CT-TH-PM surfaces, detailed simulation models may be used. However, detailed models require much development time and are computationally expensive. In this paper we present a new aggregate modeling method for simulation based calculation of workstation CT-TH-PM surfaces.

1 Introduction

A Cycle time-Throughput-Product mix (CT-TH-PM) surface of a workstation predicts the mean cycle time of products as function of the throughput of the workstation, and the product mix. Cycle time is the sum of queue time, and process time of a lot at a particular workstation. With throughput we mean the number of lots processed per time unit. The product mix is the pie chart (in percentages) of the various product types that have different process recipes. Machines may be qualified to process only a subset of product types.

CT-TH-PM surfaces may be derived from detailed simulation models representing the workstation. Yang et al. (2007) used progressive model fitting to derive CT-TH-PM surfaces from a detailed simulation model. Simulation approaches allow the inclusion of many details of the factory floor to arrive at accurate CT-TH-PM surface prediction. However, simulation models may involve considerable development and maintenance time, and it may not be possible to measure all required input parameters. In this paper we present an alternative simulation approach to obtain CT-TH-PM surfaces of a workstation. Modeling of various details is avoided by introducing so-called effective process time (EPT) distributions for each machine in the workstation.

Hopp and Spearman (2000) defines the EPT as 'the process time seen by a lot from a logistical point of view'. Hopp and Spearman combine raw process time, preemptive, and non-preemptive outages to compute the first two moments of workstation EPT distributions. Jacobs et al. (2003) calculate EPT distributions from simple events from the factory floor, being arrivals and departures of lots on workstations, without quantifying the contributing factors. Jacobs et al. use the mean and variance of the measured EPT distribution in a G/G/m queueing approximation (see e.g. Hopp and Spearman 2000; Sakasekawa 1977; Whitt 1993) to calculate cycle time-throughput (CT-TH) curves. The G/G/m queuing approximation assumes that machines process one lot at a time.

Kock (2008) developed an EPT-based aggregate modeling method to calculate CT-TH curves for workstations with integrated processing machines. An integrated processing machine may process multiple lots at the same time in the various machine chambers. The aggregate model is again a G/G/m type of approximation but now with work-in-progress dependent process times. Veeger et al. (2008) show that the EPT-based aggregate simulation models obtained using the method of Kock (2008, pp. 79-96) provide accurate CT-TH curves for workstations in a semiconductor environment. However, the aggregation methods of Kock (2008), as well as the method of Jacobs et al. (2003) are unable to predict the mean cycle time as function of the product mix.

A new EPT-based aggregate modeling method is presented in this paper, that is able to calculate CT-TH-PM surfaces for workstations consisting of one or more integrated processing machines. Our proposed aggregate simulation model consists of an infinite buffer, and several parallel servers. Each server is qualified for certain recipes. The process time of a lot at an aggregate server is sampled from an EPT distribution. The EPT-distribution parameters depend on the momentary workload, and the process recipe of the lot. Similar to Kock (2008) and Jacobs et al. (2003) we measure EPT distributions from lot arrivals and departures, which incorporate raw process time, machine outages, and setup time etc. However, we do not incorporate in the EPT time losses due to machine qualification. In the aggregate model, the product mix and recipe qualification of the servers is modeled explicitly.

The outline of the paper is as follows. In Section 2, we explain the concept and use of a CT-TH-PM surface. The developed EPT-based aggregate modeling method is presented in Section 3. Next, we test our method on two test scenarios in Section 4. Finally, we present our conclusions in Section 5.

2 Cycle time-Throughput-Product Mix Surfaces

An example of a contour plot of a CT-TH-PM surface is given in figure 1. Figure 1 gives the cycle time (indicated by the contour lines) as function of the throughput δ and the percentage p_A of lots requiring recipe A. The system considered in figure 1 is a three machine workstation that processes lots requiring either process recipe A, or process recipe B. Two of the three machines can only process recipe A, whereas the third machine can only process type B. The machines process one lot at a time. The process times are gamma distributed with a mean of 3.0 and coefficient of variability of 1.0.

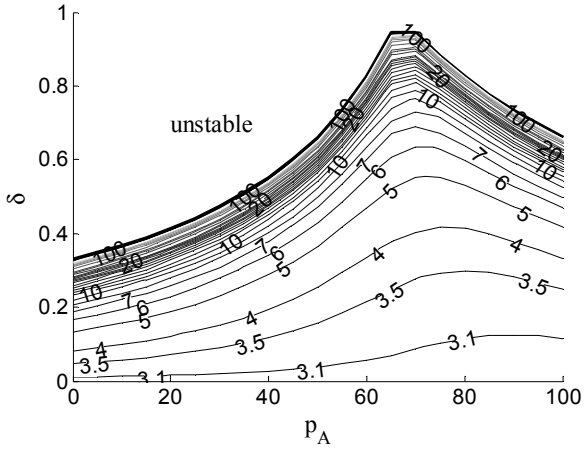

Figure 1: *Example of a CT-TH-PM Surface*

Figure 1 shows that for low throughput levels, the cycle time approaches the mean process time (3.0). For increasing throughput, lots experience queuing and the mean cycle time will increase. The value of δ for which the system becomes unstable (cycle time will go to infinity) depends on the product mix. If $p_A = 0$, only recipe B lots arrive. Since recipe B lots can only be processed at one machine, the maximum throughput equals the maximum throughput of machine 3, which is 0.33. If $p_A = 1.0$, only recipe A lots arrive, which can be processed on two out of three machines. Hence, the maximum throughput is 0.67. The maximum throughput is realized for $p_A = 0.67$, when all machines have an equal amount of workload on average.

3 EPT-based Aggregate Modeling Approach

Consider a workstation that consists of multiple parallel machines. Each machine is a multiple-process type of machine, which has one or multiple process chambers. This means that the machines can have one or more than one lot in process at the same time. Let R be the total set of process recipes r that can be processed by the machines in the workstation. Each machine m in the workstation is qualified for a subset $\rho_m \subseteq R$. We assume the buffer feeding the station is infinite and a lot can always leave the system (no blocking after service). An example of such a system is a workstation with three parallel cluster tools as shown in figure 2(a).

3.1 Model Concept

The aggregate model we use to approximate the system is shown in figure 2(b). The model consists of M parallel servers with M equal to the number of machines in the real workstation. Each aggregate server in the model corresponds to a machine and is qualified for the same subset of process recipes as its real counterpart. In the model it is assumed that lots arrive according to some arrival process in an infinite first-in-first-out (FIFO) buffer that feeds the parallel servers. Furthermore, in the model we assume that service starts if a machine is, or becomes, idle and lots are

present in the system for which the machine is qualified. The process time of a lot is sampled from a distribution at the moment it starts processing in the aggregate model.

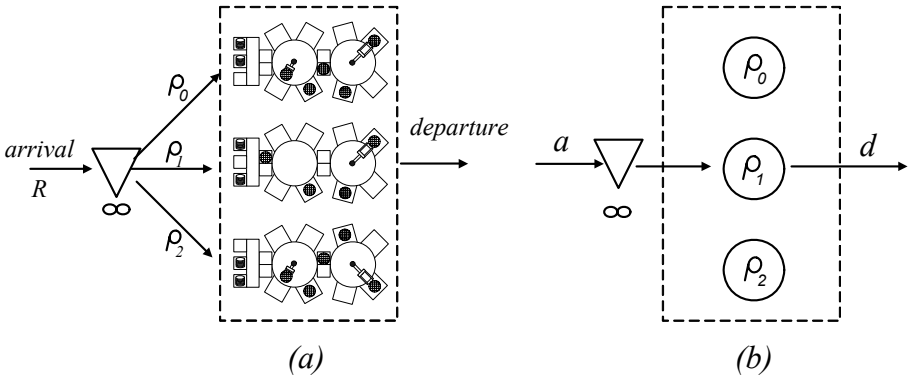

Figure 2: *Workstation with three Parallel Cluster Tools, with each Tool Qualified for a Subset of Process Recipes (a), and the Proposed Aggregate Model (b)*

Let w_i be the number of lots in the aggregate system at the process start of lot i for which the aggregate server serving lot i is qualified (including lot i itself). Let r_i be the process recipe of lot i. Let sev_i be the event type (arrival or departure), upon which lot i starts being served in the aggregate model: sev_i is 'A' if lot i starts processing on an aggregate server immediately upon arrival in the system (which happens if an idle server is qualified for lot i); sev_i is 'D' if lot i starts upon departure of another lot (i.e. lot i was queued before starting service).

We furthermore define process time buckets: each bucket is identified by event type *sev*, recipe r, number of lots w. In the aggregate model, the process time of lot i is sampled from a gamma distribution with distribution parameters (mean and variance) corresponding to bucket (r_i, w_i, sev_i). We use an independent process time distribution for each bucket. In theory w_i may be infinite, hence the number of different buckets may be infinite. However, above a certain value of w the behaviour of the process time distribution stays approximately the same since the system is already operating at its full throughput. To limit the number of buckets, we define a highest value of w being N. Buckets (r, N, sev) contain all process times of buckets (r, w, sev) for $w \geq N$.

The input of the model consists of EPT distributions for each bucket. To determine these EPT distributions, arrival and departure data is used. For each lot i departing from the considered 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. The arrival and departure data is translated into EPT-realizations using an EPT algorithm.

3.2 EPT Algorithm

We propose the algorithm shown in figure 3 to calculate EPTs. In the algorithm, we suppose that the real system from which the arrivals and departures were obtained

behaves according to our aggregate model representation. An EPT realization ends if a lot departs. The start of an EPT realization may occur in either of two cases:

1. A lot arrives while, according to the aggregate model, there is an idle machine qualified for the recipe of the lot.
2. A lot departs from a machine and, according to the aggregate model, at least one lot is waiting in the queue for which the machine is qualified.

```
loop
    read τ, id, ev, r
    if ev = 'A' then
        rs := append(rs, r)
        qs := find(as, r, qual)
        if qs = [] then                                          (1)
            ws := append(ws, (id, r))
        else                                                     (2)
            m := select(qs, mcrit)
            as := remove(as, m)
            w := count(rs, m, qual)
            ss := append(ss, (τ, id, m, r, w, ev))
        endif
    elseif ev = 'D' then
        rs := remove(rs, r)
        if id ∈ ids(ss) then                                     (3)
            (t, id, m, r, w, sev) := get(ss, id)
            ls:= find(ws, m, qual)
            if ls = [] then
                as := append(as, m)
            else
                (nid, nr) := select(ls, lcrit)
                ws := remove(ws, nid)
                nw := count(rs, m, qual)
                ss := append(ss, (τ, nid, m, nr, nw, ev))
            endif
        else                                                     (4)
            (t, nid, m, nr, w, sev) := select(ss, r, qual, scrit)
            ss := remove(ss, nid)
            nw := count(rs, m, qual)
            ss := append(ss, (τ, nid, m, nr, nw, ev))
            ws := remove(ws, id)
        endif
        write τ – t, r, w, sev
    endif
end loop
```

***Figure 3**: EPT Algorithm*

The input of the algorithm consists of events, each event represented by event time τ, lot identifier (lot ID) *id*, event type *ev*, and lot recipe *r*. The events are sorted on increasing time τ. There are *M* parallel servers in the workstation. The algorithm has a user-defined function 'qual', in which the recipe qualification of each server is defined.

In the algorithm, the following variables are used: *as* is a list that contains the IDs of the machines that are idle. Initially, *as* contains all machine numbers (1,...,*M*). Variable *ss* is a list containing EPT starts and is initially empty. An EPT start is a tuple containing time τ, lot ID *id*, aggregate server ID *m* on which the EPT started, lot recipe *r*, number of lots *w*, and event *ev*. Recall that *w* represents here the number of lots in the system upon the EPT start for which machine *m* is qualified. Variable *ws* is a list containing lots that have arrived, but that have not yet started an EPT (i.e. are waiting according to the aggregate model). List *ws* is initialized as an empty list. Each waiting lot is represented by a tuple consisting of lot ID *id* and lot recipe *r*. List *rs* contains the process recipes of all lots present in the system, and is initialized as an empty list. List *qs* is filled with machine IDs of idle machines qualified for recipe *r*, and list *ls* is filled with lot IDs for which a machine *m* is qualified. Variables *nid, nw,* and *nr* represent respectively the lot id, number of lots, and recipe of a new lot for which a new EPT is started upon departure of a previous lot. Variable *sev* represents the event type ('A' or 'D') upon which the EPT started.

We use functions find, qual, select, get, count, ids, remove, and append. Function find can determine: all machines in idle machine list *as* that can process recipe *r*, or alternatively all lots in wait list *ws* for which machine *m* is qualified. Function find uses function qual, which returns a list of recipes for which machine *m* is qualified. Function select returns one element of a list, according to a certain criterion (mcrit, lcrit, or scrit as explained later). With function get, the tuple in *ss* corresponding to lot *id* is returned. Function count counts the number of lots in list *rs* that can be processed on machine *m*, using function qual. Function ids returns all lot IDs in the tuples of list *ss*. Function append appends an element to the end of a list. Function remove removes the tuple corresponding to lot *id* (or *nid*) from an input list.

The algorithm distinguishes four cases:

1. A lot arrives and there is no idle machine qualified for the recipe of the lot *r* (*qs* is empty). Lot id *id*, and recipe *r* are added to wait list *ws*.

2. A lot arrives and there are idle machines qualified for lot recipe *r*. A machine *m* is then selected from *qs* according to criterium mcrit, and then removed from *as*. Next, the number of lots *w* for which machine *m* is qualified is determined with function count. Subsequently, an EPT start represented by tuple (τ, *id, m, r, w, ev)* is added to EPT start list *ss.*

3. A lot with id *id* departs and information of the lot is present in one of the tuples in list *ss*. This information is retrieved using function get. Function find selects all lots from wait list *ws* that can be processed on the machine that just became idle, machine *m*. If there are no lots in the queue that can be processed on machine *m* (*ls* is empty), machine *m* is added to idle machine list *as.* Else, a lot to be processed at machine *m* is selected from *ls* according to criterium lcrit, and is then removed from wait list *ws*. The new number of lots *nw* for which machine *m* is qualified is determined using function count, and tuple (τ,

nid, m, nr, nw, ev) is added to EPT start list *ss*. Herein, *nid* is the lot id of the next lot to be processed, *nr* its recipe.

4. A lot departs and lot id *id* is not known in start list *ss*. This may occur if a lot overtakes several other lots. Then, function select chooses an EPT start of another lot that started an EPT, according to criterium scrit. Criterium scrit requires recipe *r*, and function qual to select an EPT start. The EPT of the lot from which the EPT start has been used by lot *id* is then restarted: the entry containing lotID *nid* is removed from EPT start list *ss*. Then *nw* is determined, the new start tuple (*τ, nid ,m, nr, nw, ev)* is added to list *ss*, and the remaining entry of lot ID *id* in list *ws* is removed.

In the algorithm, three criteria are used with function select: mcrit, lcrit, and scrit. Criterion mcrit operates on the list *qs* of idle machines qualified for recipe *r*. Criterion lcrit operates on list *ls* of lots for which a machine *m* is qualified. Criterion scrit operates on EPT start list *ss*. Considering mcrit, in this paper we choose the machine from list *qs* that is idle for the longest amount of time (which is the first element). For lcrit, we pick the first arrived lot from *ls*. For scrit, we randomly choose an EPT start from all lot starts in *ss* having the same recipe *r* as the departing lot. If there is no EPT start available with recipe *r*, an EPT start is chosen randomly from all EPT starts that could have started on the machine where the departing lot was processed.

After each lot departure, the algorithm saves the EPT (equal to τ - t), adding variables *r, w, sev,* which are used to assign the EPT to the correct bucket. Then, the EPT distribution parameters, which are mean EPT t_e and coefficient of variability $c_{e,}$ are calculated for each bucket (*r, w, sev)*.

4 Simulation Test Case

We tested the aggregate modeling approach on the simulation test case system presented in figure 4(a). The test case system is a three machine workstation, in which each machine consists of two sequential integrated processes, each having gamma distributed process times. Two machines can process recipe A and B, whereas the third can only process recipe B. Two test scenarios are considered. In the first test scenario, both recipes have equal process time distributions, whereas in the second scenario both recipes have different process time distributions. The aggregate model we use to approximate the test case system is depicted in figure 4(b).

Simulation results were generated with the specification language χ1.0 (Hofkamp and Rooda 2007). The maximum throughput of the real-life model δ_{max} has been simulated using one simulation run of 10^5 lots. For this simulation experiment, lots are always available in the buffer. The mean cycle time has been calculated for the real-life models, as well as for the aggregate models we use to approximate the real-life models. In order to determine the mean cycle time, 30 replications of simulations of $1 \cdot 10^5$ lots are performed. For each replication, the mean cycle time of lot $2 \cdot 10^4$ - $1 \cdot 10^5$ is calculated. The total mean cycle time is determined by averaging the mean cycle time of each simulation replication.

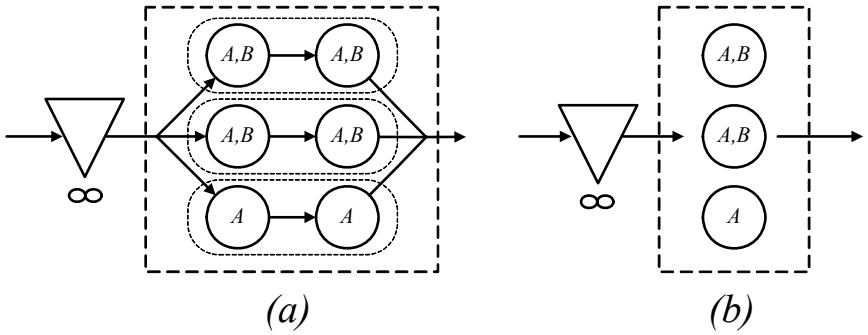

Figure 4: *Simulation Test System (a), and the Used Aggregate Model (b)*

4.1 Test Case Scenarios

In the first test scenario, the process time distribution (which is gamma) of both recipes has a mean process time of 1.0 and a coefficient of variation of 0.25. The process time of the second sequential process has a mean of 2.0, and a coefficient of variability of 0.5. In the second test scenario, the process time distribution of both recipes at the first sequential process has mean 1.0 and coefficient of variation 0.25. At the second sequential process, the mean process time of recipe A lots is 2.0 and the coefficient of variability is 0.5. For recipe B lots, the mean is 4.0 and coefficient of variability is 0.5. If multiple machines are idle and qualified for an arriving lot, the arriving lot is sent to the machine of which the first sequential process is idle for the longest amount of time.

4.2 Measured EPT Distributions

To calculate EPT distributions for both scenarios, 10^6 lots were simulated in the real-life model, and arrival and departure events were obtained. Then, the EPTs have been calculated using the algorithm presented in figure 3, and assigned to buckets. For each bucket, the EPT distribution parameters have been calculated.

Figure 5 shows the mean EPT t_e as function of the number of lots w for test scenario 2. N (user defined maximum value of w) is set to 10. The EPT distribution is measured at $p_A = 0.5$, and throughput ratio $\delta/\delta_{max} = 0.8$. The left plot in figure 5 shows the mean EPT $t_{e,a}$ of EPTs that started upon arrival of a lot, for recipes A and B. The right plot in figure 5 shows the mean EPT $t_{e,d}$ of EPTs that started upon departure of a lot.

Figure 5 shows that the mean EPT for lots that started their EPT upon arrival is approximately 3 for recipe A lots, and 5 for recipe B lots. This corresponds to the mean time recipe A and B lots take to be processed by both sequential processes. For EPTs that started upon a departure, the mean approximates 2 for increasing bucket numbers for recipe A, and 4 for recipe B. This corresponds to the interdeparture time at each machine, which is equal to the process time of the second (bottleneck) sequential process. This process time is 2 for recipe A lots, and 4 for recipe B lots.

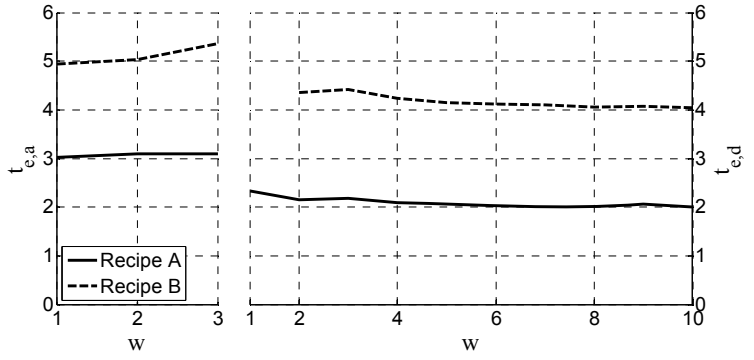

Figure 5: *Mean EPT t_e Test Scenario 2*

For EPTs of recipe A lots that started upon departure the minimum value of w equals 1. $w_i = 1$ may occur for lot i if it starts on the third machine (which can only process type A lots), after being the only recipe A lot in the queue. The third machine only 'sees' recipe A lots, hence $w = 1$. For recipe B lots, the minimum value of w equals 2. Recipe B lots can only be processed on the first two machines, which can process both recipes and therefore 'see' all lots in the system. If an EPT starts upon departure, there have to be at least two lots in the system (one queued, and one processed on the other machine that can process recipe B lots).

For test scenario 1, the mean EPTs of both recipes are approximately the same, for both recipes have equal processing times. $t_{e,a}$ is approximately 3, and $t_{e,d}$ approximates 2 for increasing w. The dotted lines lie almost on top of the solid lines in figure 5.

4.3 CT-TH-PM Curves

The EPT distributions measured at the operating point at $p_A = 0.5$ and $\delta/\delta_{max} = 0.8$ are used as input in the aggregate model shown in figure 4(b). The CT-TH-PM surface predicted by the aggregate model is compared with the model representing the real-life situation (figure 4(a)). The relative error between the cycle time estimated by the aggregate model, and the real cycle time $e(p_A,\delta)$ is defined as:

$$e(p_A,\delta) = \frac{\hat{\varphi}(p_A,\delta)}{\varphi(p_A,\delta)} - 1, \tag{1}$$

where $\hat{\varphi}(p_A,\delta)$ is the mean cycle time estimated by the aggregate model and $\varphi(p_A,\delta)$ is the cycle time calculated by the real-life model.

Figure 6 shows the results for test scenario 1 and 2. The left part of figure 6 shows the cycle time of the real-life model, and the cycle time predicted by the aggregate model as function of the throughput δ for various values of the product mix p_A. The right part of figure 6 shows the relative error e as function of δ/δ_{max} for various values of p_A.

Figure 6 shows that the relative errors for test scenario 1 are within -0.1 and 0.1 for throughput levels δ/δ_{max} between zero and approximately 0.95. For test scenario 2,

accuracy is slightly less, in particular for p_A values other than $p_A = 0.5$ for which the model is trained. At $p_A = 0.5$, the EPT-distributions of recipe B lots are influenced by recipe A lots, and the other way around. For p_A near 0.0 or near 1.0, the mutual influence is much less present, which is not taken into account in the model. Nevertheless, the predicted CT-TH-PM curves are still very close to the real ones.

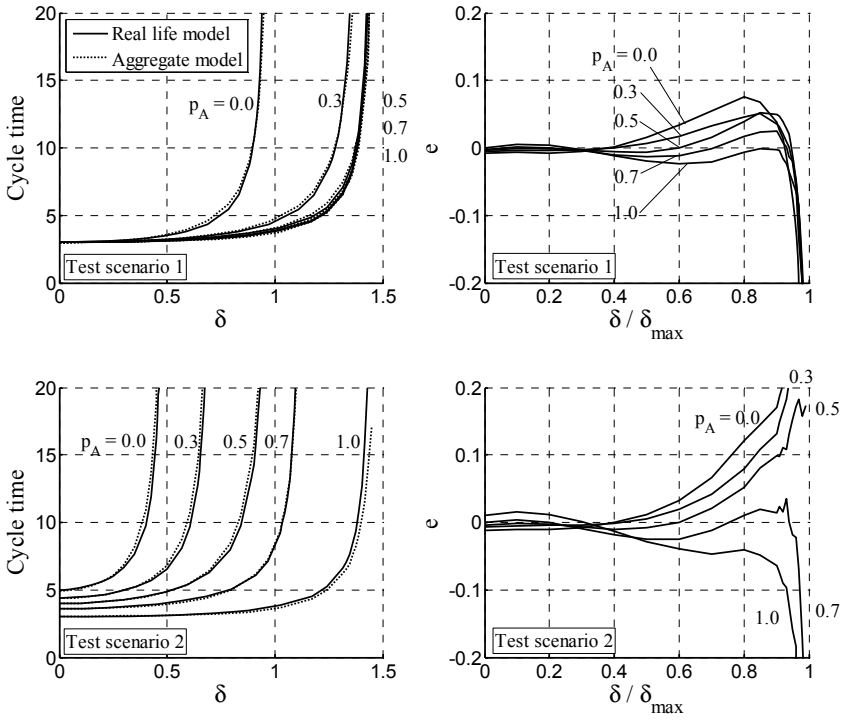

Figure 6: *Real and Estimated Mean Cycle Time, and Relative Error of Test Scenario 1, and 2*

5 Conclusion

In this paper, an EPT-based aggregate modeling method is presented to calculate CT-TH-PM surfaces of workstations with integrated-process type of machines. Machines may be qualified for a subset of process recipes. The new method approximates the workstation by an *m*-server parallel workstation, in which each server is qualified for the same set of recipes as in the original system. In the aggregate m-server workstation each server processes just a single lot at a time. The process time distribution of the servers depends on the lot recipe, workload, and the condition on which the lot starts processing. The process time distributions are measured directly from lot arrivals and departures using an EPT algorithm. The proposed EPT-based aggregate modeling method is tested on two test scenarios. For test scenario 1, accurate results were obtained for throughput ratios below 0.95 for all product

mixes. For test scenario 2, accurate results were obtained for product mix around the operational point of the system.

In future work, we aim to test the proposed method on a simulation testcase based on a semiconductor workstation. This simulation testcase consists of several cluster tools in parallel. Each cluster tool is modeled in detail, and is qualified for certain recipes. After that we will test the method on a real semiconductor workstation.

References

Hofkamp, A.T.; Rooda, J.E. (2007) Chi 1.0 Reference manual. Systems Engineering Group, Eindhoven University of Technology, http://se.wtb.tue.nl/sewiki/chi/

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

Jacobs, J.H.; Etman, L.F.P.; Campen, E.J.J.v.; Rooda, J.E. (2003) Characterization of operational time variability using effective process times. IEEE Transactions on Semiconductor Manufacturing, 16(2003) 3, pp. 511-520

Kock, A.A.A. (2008) Effective process times for aggregate modeling of manufacturing systems. PhD Thesis, Eindhoven University of Technology

Sakasekawa, H. (1977) An approximation formula $l_q = \alpha\beta^{\rho}/(1-\rho)$. Annals of the Institute for Statistical Mathematics, 29, pp. 67-75

Veeger, C.P.L.; Etman, L.F.P.; Herk, J.v.; Rooda, J.E. (2008) Generating cycle time-throughput curves using effective process time based aggregate modeling. In: Proceedings of the 2008 Advanced Semiconductor Manufacturing Conference (ASMC), Boston (USA). CD-ROM publication, pp. 127-133

Whitt, W. (1993) Approximating the GI/G/m queue. Production and Operations Management, 2(1993) 2, pp. 114-161

Yang, F.; Liu, J.; Tongarlak, M.; Ankenman, B.E.; Nelson. B.L. (2007). Metamodeling for cycle time-throughput-product mix surfaces using progressive model fitting. In: Henderson, S.G.; Biller, B.; Hsieh, M.-H.; Shortle, J.; Tew, J.D.; Barton, R.R. (eds.) Proceedings of the 2007 Winter Simulation Conference, Washington D.C. (USA). IEEE Press, New Jersey, pp. 322-330