Proceedings of the 2016 Winter Simulation Conference
T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick, eds.

A DEMONSTRATION OF MACHINE LEARNING FOR EXPLICIT FUNCTIONS FOR CYCLE TIME PREDICTION USING MES DATA

Birkan Can
Cathal Heavey

Enterprise Research Centre
University of Limerick
Limerick
IRELAND

ABSTRACT

Cycle time prediction represents a challenging problem in complex manufacturing scenarios. This paper demonstrates an approach that uses genetic programming (GP) and effective process time (EPT) to predict cycle time using a discrete event simulation model of a production line, an approach that could be used in complex manufacturing systems, such as a semiconductor fab. These predictive models could be used to support control and planning of manufacturing systems. GP results in a more explicit function for cycle time prediction. The results of the proposed approach show a difference between 1-6% on the demonstrated production line.

1 INTRODUCTION

Cycle time is one of the key metrics for operating the industrial systems (Pfund, Mason, and Fowler 2006). In manufacturing, cycle time estimates allow scheduling and due-date assignment applications (Chiang 2013, Sha and Hsu 2004). However, predicting cycle time for a factory is not straightforward, particularly in complex factories, such as those in semiconductor manufacturing. Process flow may include hundreds of process steps, re-entrant routes, equipment failures and batching.

In literature, queueing models and detailed factory simulations have found uses in predicting cycle times. Evaluating queueing models rapidly provide estimations; while, representing a complex manufacturing systems can lead to intractable models (Shanthikumar, Ding, and Zhang 2007). Discrete-event simulation on the other hand can realistically capture the factory complexities; however, the benefits are often realised at the expense of model development and maintenance, and evaluation time (Pfund, Mason, and Fowler 2006).

Abundance of system data in real-time in today's factories and the advances in data analytics motivated machine learning studies for predicting cycle times and other performance metrics such as throughput (Can, Heavey, and Kabak 2014, Liao, Chu, and Hsiao 2012, Monostori 2003).

This paper postulates that genetic programming, a machine learning technique, can be used to develop predictive models of process cycle times based on system status information gathered in real-time. Typically, the required data can be extracted from a manufacturing execution system (MES). To demonstrate the approach we use a simulated model of a production line, which will generate work-in-process (WIP), WIP distributed dynamically in the line and EPT.

In the remainder of this article, we first provide a literature review in Section 2 of similar approaches. Then the proposed approach is presented in Section 3 detailing the components of the framework. Following, the results from an empirical analysis demonstrating the approach is provided in Section 4 to motivate the discussion in Section 5.

2 LITERATURE REVIEW

In studies concerned with cycle time prediction in complex manufacturing systems, such as semiconductor fabs, several applications can be found. Briefly, the idea is to group historical data based on a similarity or distance measure; and then to predict cycle time of a future system state based on historical data. Backus, Janakiram, Mowzoon, Runger, and Bhargava (2006) use historical information on process parameters such as work-in-progress at specific operations, lot priority and product type to compare clustering, K-nearest neighbours, and regression trees. The study reports that the regression trees based on recursive binary rules (e.g., if-then-else) perform comparably better than other approaches, even with the cases where outlier observations were kept.

There have been recent studies using Artificial Neural Networks (ANNs) with fuzzy or clustering components reporting promising results in predicting cycle times (Chen 2003, Chen, Jeang, and Wang 2008). Chang, Fan, and Wang (2009) considered the cycle time observations as time series data to predict lot output times in wafer fabs using ANNs. Chang, Fan, and Wang (2009) enhanced ANNs with fuzzy rules to incorporate domain knowledge in the form of logic rules (i.e., if-then-else) similar to Chen (2003). Chen, Jeang, and Wang (2008) presented a hybrid of ANN, Self-Organizing Map (SOM) and Case Based Reasoning (CBR). In follow on work, Chen and Lin (2009) incorporated fuzzy c-means to classify historical wafer information in a semiconductor manufacturing plant. Chen and Lin (2009) compared their method against exponential smoothing and previous ANN applications. Their approach (Chen and Lin 2009) achieves an average of 38% higher performance compared to previous ANN-based algorithms and they observed that the prediction error grows with increasing workload level fluctuation.

Recently, Meidan, Lerner, Rabinowitz, and Hassoun (2011) presented a thorough analysis to identify queue (waiting) time factors and subsequently developed ANNs to predict queueing times. The study first outlines over 180 factors from a semiconductor manufacturing factory. At the end of the feature selection process, Meidan, Lerner, Rabinowitz, and Hassoun (2011) reported that the number of parameters was reduced down to 47. Subsequently, different configurations of the designed framework was tested on shop-floor data to investigate the performance.

There are also rare applications of other kernel-based learning methods as an alternative to ANNs in cycle time estimation. Alenezi, Moses, and Trafalis (2008) exemplify the use of support vector regression (SVR) on multi-resource multi-product manufacturing systems of varying complexity. In the study, SVR predictions are compared against exponential smoothing, moving averages and ANNs. The results report a success rate with SVR of $13-23\%$ in terms of root mean squared error (rmse) outperforming both ANNs and forecasting via moving averages. Despite the generalisation ability of SVRs, they have found relatively rare application (Li, Ng, Xie, and Goh 2010).

Baykasoğlu and Göçken (2009) used a variant, gene expression programming (GEP) to derive a due-date assignment strategy. This is done by estimating the completion times using process variables in multi-stage job shops. The due-date assignment strategy with closest performance to GEP, referred to as ADRES, exploits the sum of the most recent queue waiting times at each station, transportation time to each station and the average raw process time for the job. Chiang (2013) explores scheduling applications of evolutionary algorithms and GP and provide an extensive review of means of developing and ranking scheduling rules in semiconductor manufacturing. Can, Heavey, and Kabak (2014) used GP to model operating curves of a detailed discrete event simulation (DES) of a wetbench. The functions found where of an explicit mathematical form with the ability to change inputs to predict changes to the operating curve.

The applications considering data mining and machine learning are generally carried out in the framework of knowledge discovery in databases. In an extensive review, Choudhary, Harding, and Tiwari (2009) outline 10 steps of the process of developing models for studying system behaviour. Comparison between the different models is difficult due to the different system or segments of system that the models where applied to.

Jacobs, Etman, Van Campen, and Rooda (2003) demonstrated how the (overall) impact of system realities on performance can be estimated using effective process time (EPT) distributions. This illustrates a means for overcoming the aforementioned challenges of the diversity of the characteristics of a system by using concepts from factory physics. EPT is defined as the capacity claimed by the products and accounts for the irregularities, such as, machine failures, rework, maintenance, set-ups, batching and so on (Jacobs, Etman, Van Campen, and Rooda 2003). This definition of EPT provides a convenient way to characterise system behaviour.

The above provides a brief review of predictive cycle time models. The literature review indicates that there are several possible challenges including the variety of approaches, the amount of information required and the selection of appropriate characteristic system data. For this reason, use of aggregate metrics instead of a detailed parametrisation could be advantageous for cycle time prediction purposes. In the next section, the method of using work-in-process levels, EPT and GP for this purpose is presented.

3 THE PROPOSED APPROACH

In this paper, we demonstrate this approach using DES of a production line as an illustration of how the framework would operate. In a real system, Figure 1 illustrates how the framework would integrate with an MES of a manufacturing system (i.e. a semiconductor fab) which would gather data on a job's (i.e., product, lot) arrival to and release events.

More specifically, three state variables are used to define the system state: (i) current work-in-progress (WIP_t^k); (ii) current queue lengths of the process buffers, e.g., WIP profile or component WIP, ($\overrightarrow{WIP_t^k}$) and (iii) the average of the effective process times (EPT) of lots in the system averaged over the queue time of a job k in queue prior to its release to the system (EPT^k). Let i_j^t equal the current inventory (contents of buffer and server) at station j at time t, then $\overrightarrow{WIP_t^k}$ is defined by the vector $(i_1^t, i_2^t, \ldots, i_m^t)$, where t equals the time of arrival of lot k and $WIP_t^k = \sum_{j=1,\ldots,m} i_j^t$. This data includes state variables, WIP_t^k,$\overrightarrow{WIP_t^k}$ and EPT^k seen by a job and incurred cycle time of the job as the target variable. After several evolutionary processes, GP derives cycle models representative of the system behaviour abstracted by the state variables.

In parallel to the development process, the constructed predictive models are validated on unseen data. In this paper this is done by running new replications of the DES model. In the real system this would be done by selecting the appropriate period to train the GP and a period of time to validate the GP. For periods, where the decision rules in the system remains unchanged (e.g., dispatching rules), these models can be used for analysis and control, for instance, the line loading and order releases. The real-time data acquired from a system for a job can be evaluated to estimate its cycle time with the symbolic regression model developed by GP using recent historical information on the aforementioned metrics.

The amount of WIP in a manufacturing system is mainly driven by utilisation (amount of loading in the system). However, total WIP alone may not be enough to represent the system state; per cent completion of the intermediate products in the system is not available. On the other hand, this level of information could be inferred from queue lengths. Therefore, we considered the queue lengths, in other words, the WIP profile. While WIP_t^k represents the total WIP in the line in front of a lot, $\overrightarrow{WIP_t^k}$ corresponds to the distribution of the inventory across the line, i.e., inventory levels at the stations (including queue and server) at the time of a lot's arrival to the system. In this manner, the latter is a more granular (higher granularity) representation of WIP_t^k; and hence, provides more detailed information.

The literature suggests that EPT can provide a means to assess the impact of such disturbances (Hopp and Spearman 2000). Sample path equation (Equation 1), with a_i and d_i equal to the arrival and departure times of a lot, respectively, can be used to quantify EPTs from lot arrival and departure events in the case of FIFO processing (Buzacott and Shanthikumar 1993). We will adopt this quantification of EPT for the considered manufacturing scenarios.

$$EPT_i = d_i - max(a_i, d_{i-1}) \tag{1}$$

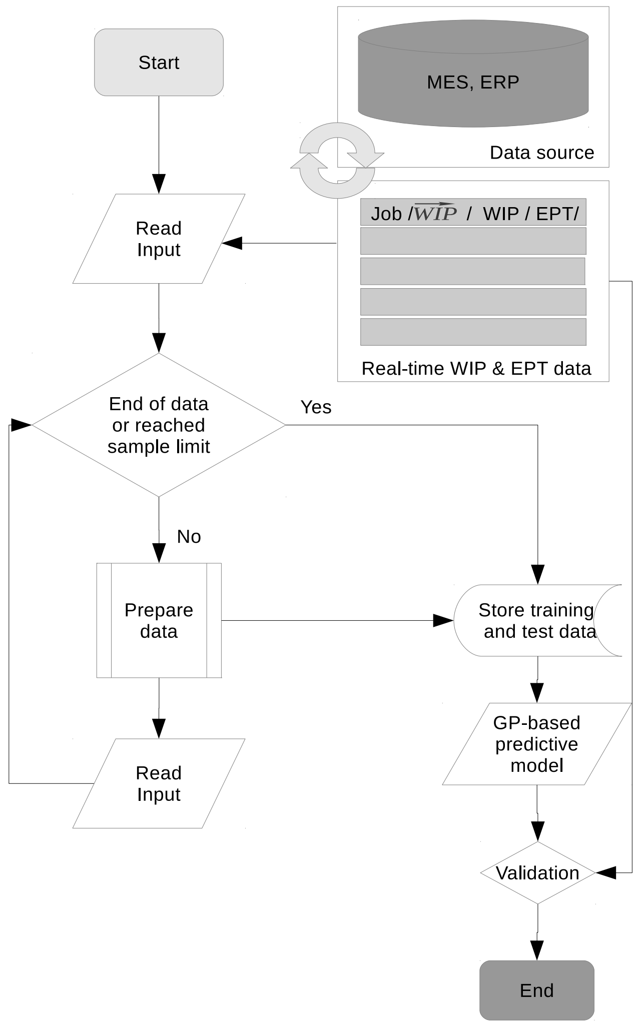

Start
MES, ERP
Data source
Job /$\overrightarrow{WIP}$ / WIP / EPT/
Real-time WIP & EPT data
Read Input
End of data or reached sample limit
Yes
No
Prepare data
Store training and test data
Read Input
GP-based predictive model
Validation
End

Figure 1: Using GP, WIP profile, and historical performance quantified by EPT in real-time data collection system.

```
1:  input: genetic programming configuration, training data;
2:  output: best expressions;
3:  begin
4:  var gen:=1, eval :=0, pop:=0, newPop, currentPop;
    Randomly create initial population from primitives supplied;
    Execute each expression on training data for fitness;
5:  set currentPop:= Initial population
6:  while gen< Max Generation and eval < Evaluation Limit
7:      while pop ≤ Population Size and eval < Evaluation Limit
8:      Select two parents from the currentPop;
9:      Create new offspring via crossover;
10:     Insert the offspring into newPop;
11:     set pop:= pop+1, eval:= eval+1;
12:     end while
13: set currentPop = newPop, gen:= gen+1;
14: end while
15: return best solution so far;
16: end
```

Algorithm 1: The general framework of a standard GP algorithm.

Determining the period over which an EPT is captured may be challenging in real world scenarios due to structural disturbances (e.g. product mix, scheduled maintenance) and the need to include system realities. Readers are referred to Kock, Etman, and Rooda (2008) and Veeger, Etman, van Herk, and Rooda (2010) which provide extensions to algorithms developed for prediction of EPT distributions based on WIP levels in various scenarios. Kock, Etman, and Rooda (2008) test the EPT algorithms in different workstations with single or multiple servers and re-entrant products to produce operating curves under different loading schemes. Veeger, Etman, Lefeber, Adan, Herk, and Rooda (2011) use EPT algorithms, which read arrival and departure information of a simulated or a real system, to calculate EPTs and record the corresponding WIP content of the whole system. For each WIP level, an EPT distribution is fitted to generalised distributions, such as, Gamma distribution. Subsequently, (WIP, EPT) pairs are used to simulate an aggregate G/G/m system as a black-box and predict cycle time distributions. Equation 1 can be replaced by algorithms from Kock, Etman, and Rooda (2008) and Veeger, Etman, Lefeber, Adan, Herk, and Rooda (2011).

In the experiments presented here, an average EPT is measured using sample path equations (Equation 1) based on the average of the EPTs collected over the lot's arrival to the system to the time the job starts processing on the first station. This instant is the release of the job to the system and determined by scheduling/dispatching. Therefore, the maximum possible number of EPT observations used would be the WIP capacity of the system minus 1. Assume that lot k arrives at time t to the system and lot k spends duration h in the first queue, then:

$$EPT^k = Average\left(EPT_{i|_t^{t+h}}\right), \tag{2}$$

where $EPT_{i|_t^{t+h}}$ refers to the EPT_i realisations made from time t to time $t+h$.

3.1 Genetic Programming

GP (Koza 1992) is an evolutionary algorithm which is capable of discovering relationships underlying a data set (Eggermont et al. 2005). The data set may belong to a system and includes a set of decision variables (input) and corresponding values for a performance metric (output) of interest.

The pseudo code for a GP is provided in Algorithm 1. Common to EAs, GP works on a set of individuals, i.e., population, by iterative application of evolutionary mechanisms, e.g. crossover, selection.

Table 1: Manufacturing line scenarios implemented in DES considering process blocks with different characteristics.

Parameters	Experiment ID								
	10	20	30	21	31	22	32	23	33
Arrival rate, λ	1	1	1	1	1	2	2	1	1
Servers, μ	1	1	1	2	2	4	4	4	4
Buffer sizes, N	2	4	7	4	7	4	7	4	7

A fitness function (Equation 3) based on the error measure, mean absolute error (*mae*), (see Equation 4) is preferred over the root mean squared error since the use of squared terms can amplify the error from outliers, undermining the regression performance (Pynnönen and Salmi 1994, Dasgupta and Mishra 2004, Can 2011). Equation 4 presents the formula used to calculate the *mae* value, where n is the size of the training data set, y_i is the output for the i^{th} observation and $\hat{y}_i$ is the corresponding prediction made by the function developed.

$$fitness = \frac{1}{1+mae}\% \tag{3}$$

$$mae = n^{-1} \sum_{\forall i \le n} |y_i - \hat{y}_i| \tag{4}$$

As a consequence of the flexibility of the evolving system and robustness of the algorithm, there have been a number of symbolic regression applications in a range of domains; clustering and data mining applications (Eggermont et al. 2005), modelling of chemical (McKay, Willis, and Barton 1997) and electronics manufacturing processes (Chan, Kwong, and Fogarty 2010) financial time series prediction (Schwaerzel 2006). Several other examples regarding manufacturing processes are provided earlier in Section 2. In the next section, a DES model of a production line is used to evaluate the proposed framework presented in Figure 1.

3.2 Model Description

A standard DES model of a production line is used to evaluate the proposed approach (see Figure 2). Arrival processes is assumed to follow a Poisson distribution with mean, λ and the process rates of the workstations are exponentially distributed with mean, μ. Buffer levels are capacitated in a fashion to allow the system to experience starvation and blocking events (see Table 1). For example, for Experiment 32 (Table 1), the arrival rate is 2, with each workstation having a mean process rate of 4 and each buffer has a capacity of 7 products. The configurations provided in Table 1 allows a system utilisation varying in the range of 0.7-0.9, similar to Veeger, Etman, van Herk, and Rooda (2010).

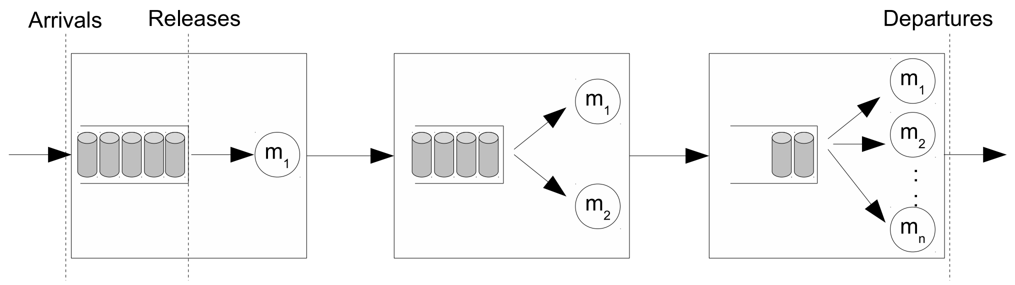


Figure 2: Example illustration of a manufacturing line with multiple process steps with single/multi-server machines, distinguishing arrivals and releases to the system and departures from it.

Table 2: The GP parameter configuration used in experiments.

Parameter	Value
Initialisation	Random, size $\in$[2,4]
Population size	400
Parent Selection	Tournaments (of 2)
Offspring Acceptance	$fitness_{offspring} > fitness_{parents}$
Crossover	Sub-tree swap

Seemingly simple, the model configurations outlined (Table 1) can imitate the stochastic behaviour of the systems subject to variation. It is important to note that, the impact of this type of irregularities are quantified by the EPT component of the proposed method and of the resultant models. So any arbitrary irregularity (e.g., setups, failures) incorporated to the system in addition will eventually be accounted by the EPT calculations. For each experiment, job information is collected during steady-state simulation post a warm-up period to populate the training and test sets. The warm-up period was derived using Welch's method and set to 20,000 jobs (Banks 1998).

Finally, for each experiment a sample size of 50 is used both for training and test of the developed cycle time approximations. To maintain a reasonable comparison between training and test results, an equal number of samples are chosen for both data sets. The input and output observations in the training and validation data are normalized for each variable based on the mean and the variance (Equation 5). The normalisation is performed in order to remove the bias of any variable due to its scale ensuring each state variable will have the same weight as an input to the algorithm.

$$x_i^{'} = \frac{x_i - \bar{x}}{Var(x)} \tag{5}$$

The next section provides the results of the application of the proposed method to the simulation models summarised above.

4 RESULTS

This section first describes the setting of the parameters used in the GP algorithm and then presents the predictive ability of the approach on the DES simulation model.

4.1 Configuration of GP

For GP to work, the components of the algorithm need to be configured. These involve setting parameters including initialisation, parent selection and information exchange (crossover) processes.

In the experiments presented here, the algorithm is configured based on the reported tests from Can and Heavey (2011). In this respect, solutions are randomly created to initialize the algorithm using expressions with sizes ranging from 2 to 4. Size regarding an expression here refers to the total number of functional and terminal parameters included in a constructed function and is a measure of complexity. We consider a population of 400 solutions (e.g., functions, predictors, metamodels) for a generation. A tree-exchange crossover is used to create offspring where the nodes to swap are randomly selected. The parent solutions are selected using tournaments of size 2 from a current generation. In this type of selection, for each parent two individuals are randomly chosen from a current population and compete with respect to their training fitness (F) (Murphy and Ryan 2008).

4.2 Predictive Performance

High accuracy of both training and test fitness of the models generated through the application of the proposed method in different manufacturing scenarios illustrates the efficacy of GP in predicting cycle time behaviour (see Table 1). In the experiments, each GP run is replicated 5 times due to the stochastic nature of the evolutionary operators. The averaged results are summarised in Figure 3 in terms of training and test fitness. In the analysis presented here, the best models are chosen from a population after which no significant increase in training performance is observed. Among solutions with similar training performance, those with fewer complexity are chosen.

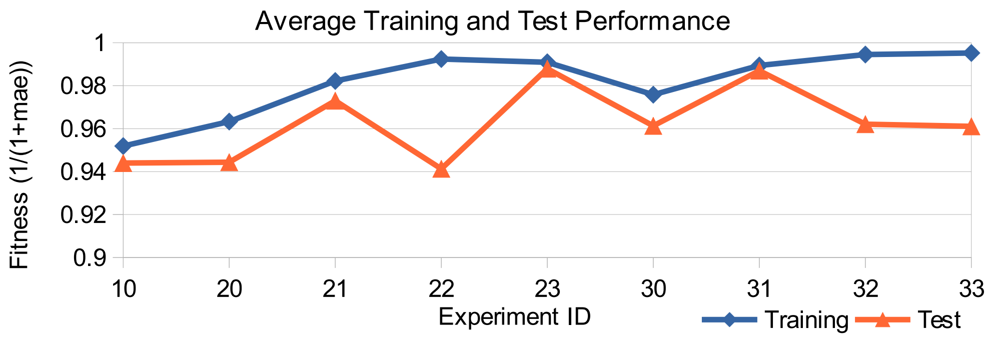
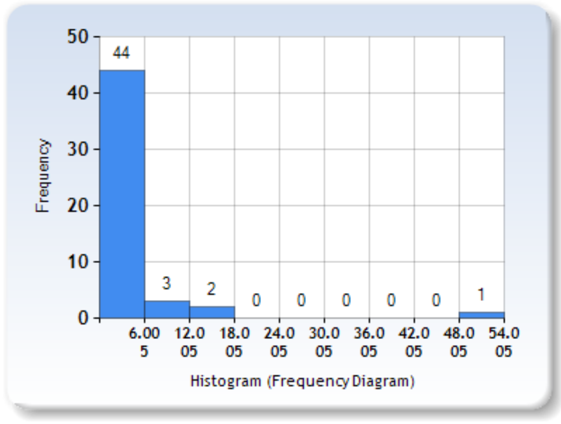
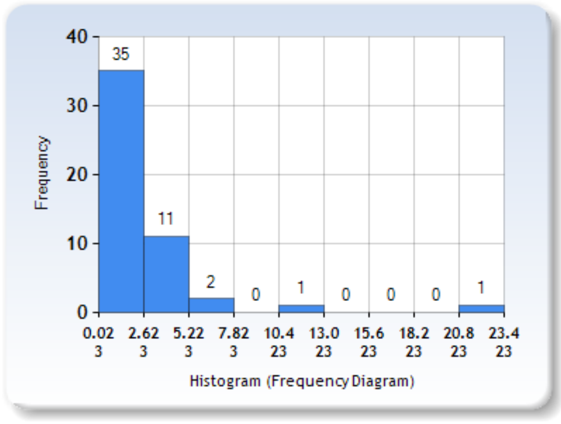


Figure 3: Average accuracy of GP metamodels in 5 replications over the studies scenarios.

The close proximity of the value of training and test fitnesses to 1 indicates the high predictive performance of GP in terms of *mae* (see Figure 3). The fitness results from both training and test are in the range of 1-6% of the actual normalised cycle times across the simulated scenarios. While overall accuracy across different system configurations demonstrate a portable approach, accuracy on unseen data implies a generalisation ability. The histograms shown in Figure 4 provide insight towards the generalisation of the developed models across the test data for the two high variability scenarios (Experiments 31 & 32). In Experiment 31, where *mae* was 3.2%, 88% of the observations had an absolute error less than 6% error and 94% within a 12% error. Similarly, in Experiment 32, where the *mae* was 2.3%, 70% were within a 3% error and 92% are within a 6% error. Smaller *mae* distributions of errors was found in all other cases.



(a) Results showing variation Exp 31.



(b) Results showing variation Exp 32.

Figure 4: Results showing variation of results for experiments Exp 31 and Exp 32.

Equation 6 exemplifies such a function from Experiment 32, where $x_i, i = 0,1,2,3$, $x_4 = WIP_t^k$ and $x_5 = EPT^k$ (see Table 3). The analytic functions obtained demonstrates an advantage of GP against its counterparts. The explicit form of the models developed provides transparency to the user (Equation 6).

$$
\begin{aligned}
CycleTime &= f_1 - f_2 && (6)\\
where, \quad & f_1 = x_5 * (x_1 - x_4) && (7)\\
& f_2 = sin(f_3 * exp(x_2 + x_3) - x_1) && (8)\\
& f_3 = x_0 - (x_2 + x_3)^9 + (x_1 * 4log(x_2)) && (9)
\end{aligned}
$$

Table 3: Predictions in comparison with the normalized target for different problem instances across the experiments.

Experiment ID	System State						Cycle Time		
	$\overrightarrow{WIP_t^k}$				WIP_t^k	EPT^k			
	x_0	x_1	x_2	x_3	x_4	x_5	Actual	Predicted	$\|error\|\%$
22	1.339	-0.613	-0.397	-0.411	-0.591	-0.034	0.042	0.043	0.024
23	0.915	0.751	0.016	-0.777	-0.155	0.052	0.057	-0.035	1.614
	0.915	-0.567	0.840	-0.391	0.031	0.064	-0.097	-0.063	0.361
31	0.604	-0.562	0.318	0.379	0.182	0.220	-0.022	-0.005	0.727
	0.604	-0.335	0.134	-0.228	-0.302	-0.040	-0.047	-0.039	0.170
32	0.525	0.499	-0.480	0.393	0.149	0.089	-0.050	-0.058	0.163
	0.525	-0.316	0.015	-0.195	-0.168	-0.013	0.134	0.029	0.782

5 DISCUSSION

In this article, GP develops cycle time prediction models using dynamic system state information. Drawing from past research on cycle time estimation and analysis, the machine learning approach exploits the positional and granular information from the simulation model used to imitate a production system.

With the approach presented, we attempted to address the challenges faced in applications of data mining and machine learning methods in the context of cycle time prediction. We identified predictive cycle time models providing a good trade-off to the exploratory analysis requirements by utilising work-in-process (WIP) profile and effective process time (EPT) as the operational metrics. Since these variables are measurable from immediate transaction data (i.e., arrival, release, departure events) and a relatively small number of observations are required, it would be possible to use this approach in real-time given sufficient computing resources.

Good predictive characteristics during both training and test indicates the portability of the approach across different scenarios. Approximate models within 1-6% accuracy range in the simulated manufacturing scenarios demonstrate that GP is robust to different system settings. On the other hand, test fitnesses being in the close vicinity (0.01 - 0.03%) of the training performance implies generalisation to unseen states of the system while highlighting that fundamental behaviour of the systems were being captured through the use of aforementioned aggregate metrics.

During the empirical analysis, certain instances on the test data illustrate the erratic cycle time behaviour which is similar to outliers in a data analysis. These also coincide with the points where the actual cycle times were larger than the predictions. In other words, there may have been a shift in the system status as a result of a combination of factors.

Although we have presented a generic framework for cycle time prediction independent of the type and the layout of the system, there are many avenues for improvement. To illustrate, increasing the granularity of information in terms of the state variables could improve the representation of the system behaviour. For instance, utilisation of the EPT calculations at the individual workstation level instead of the whole system will supply more detailed information. In such cases, dimensions of the training data will increase due to the increasing number of variables. Hence, it may be more appropriate to focus this type of investigation to a process block of a system.

REFERENCES

Alenezi, A., S. A. Moses, and T. B. Trafalis. 2008. "Real-Time Prediction of Order Flowtimes Using Support Vector Regression". *Computers and Operations Research* 35 (11): 3489–3503.

Backus, P., M. Janakiram, S. Mowzoon, C. Runger, and A. Bhargava. 2006. "Factory Cycle-Time Prediction with a Data-Mining Approach". *IEEE Transactions on Semiconductor Manufacturing* 19 (2): 252–258.

Banks, J. 1998. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. New York: John Wiley & Sons.

Baykasoğlu, A., and M. Göçken. 2009. "Gene Expression Programming Based Due Date Assignment in a Simulated Job Shop". *Expert Systems with Applications* 36 (10): 12143 – 12150.

Buzacott, J. A., and J. G. Shanthikumar. 1993. *Stochastic Models of Manufacturing Systems*. Englewood Cliffs, N.J.: Prentice-Hall.

Can, B. 2011. *Evolutionary Modelling of Industrial Systems with Genetic Programming*. Ph. D. thesis, University of Limerick, Ireland.

Can, B., and C. Heavey. 2011. "Comparison of Experimental Designs for Simulation-Based Symbolic Regression of Manufacturing Systems". *Computers and Industrial Engineering* 61 (3): 447 – 462.

Can, B., C. Heavey, and K. E. Kabak. 2014. "Generating Operating Curves in Complex Systems Using Machine Learning". In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, A. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 2404–2413. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Chan, K. Y., C. K. Kwong, and T. C. Fogarty. 2010. "Modeling Manufacturing Processes Using a Genetic Programming-Based Fuzzy Regression with Detection of Outliers". *Information Sciences* 180 (4): 506–518.

Chang, P.-C., C. Fan, and Y.-W. Wang. 2009. "Evolving CBR and Data Segmentation by SOM for Flow Time Prediction in Semiconductor Manufacturing Factory". *Journal of Intelligent Manufacturing* 20:421–429.

Chen, T. 2003. "A Fuzzy Back Propagation Network for Output Time Prediction in a Wafer Fab". *Applied Soft Computing* 2/3F:211–222.

Chen, T., A. Jeang, and Y.-C. Wang. 2008, March. "A Hybrid Neural Network and Selective Allowance Approach for Internal Due Date Assignment in a Wafer Fabrication Plant". *The International Journal of Advanced Manufacturing Technology* 36 (5): 570–581.

Chen, T., and Y.-C. Lin. 2009. "A Fuzzy Back Propagation Network Ensemble with Example Classification for Lot Output Time Prediction in a Wafer Fab". *Applied Soft Computing* 9 (2): 658 – 666.

Chiang, T.-C. 2013. "Enhancing Rule-Based Scheduling in Wafer Fabrication Facilities by Evolutionary Algorithms: Review and Opportunity". *Computers & Industrial Engineering* 64 (1): 524–535.

Choudhary, A., J. Harding, and M. Tiwari. 2009. "Data Mining in Manufacturing: A Review Based on the Kind of Knowledge". *Journal of Intelligent Manufacturing* 20 (5): 501–521.

Dasgupta, M., and S. Mishra. 2004. "Least Absolute Deviation Estimation of Linear Econometric Models: A Literature Review". MPRA Paper 1781, University Library of Munich, Germany.

Eggermont, J. et al. 2005. *Data Mining Using Genetic Programming: Classification and Symbolic Regression*. Institute for Programming research and Algorithmics, Leiden Institute of Advanced Computer Science, Faculty of Mathematics & Natural Sciences, Leiden University.

Hopp, W. J., and M. L. Spearman. 2000. *Factory Physics*. Boston, MA: Irwin/McGraw-Hill.

Jacobs, J. H., L. F. P. Etman, E. J. J. Van Campen, and J. Rooda. 2003. "Characterization of Operational Time Variability Using Effective Process Times". *IEEE Transactions on Semiconductor Manufacturing* 16 (3): 511–520.

Kock, A., L. Etman, and J. Rooda. 2008. "Effective Process Times for Multi-Server Flowlines with Finite Buffers". *IIE Transactions* 40 (3): 177–186.

Koza, J. 1992. *Genetic Programming: on the Programming of Computers by Means of Natural Selection*. Massachusetts: The MIT Press.

Li, Y. F., S. H. Ng, M. Xie, and T. N. Goh. 2010. “A Systematic Comparison of Metamodeling Techniques for Simulation Optimization in Decision Support Systems”. *Applied Soft Computing* 10 (4): 1257–1273.

Liao, S.-H., P.-H. Chu, and P.-Y. Hsiao. 2012. “Data Mining Techniques and Applications - A Decade Review from 2000 to 2011”. *Expert Systems with Applications* 39 (12): 11303 – 11311.

McKay, B., M. Willis, and G. Barton. 1997. “Steady-State Modelling of Chemical Process Systems Using Genetic Programming”. *Computers & Chemical Engineering* 21 (9): 981–996.

Meidan, Y., B. Lerner, G. Rabinowitz, and M. Hassoun. 2011. “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.

Monostori, L. 2003. “AI and Machine Learning Techniques for Managing Complexity, Changes and Uncertainties in Manufacturing”. *Engineering Applications of Artificial Intelligence* 16 (4): 277 – 291.

Murphy, G., and C. Ryan. 2008. “A Simple Powerful Constraint for Genetic Programming”. In *Genetic Programming*, edited by M. O’Neill, L. Vanneschi, S. Gustafson, A. Esparcia Alcazar, I. De Falco, A. Della Cioppa, and E. Tarantino, Volume 4971 of *Lecture Notes in Computer Science*, 146–157. Springer Berlin / Heidelberg.

Pfund, M. E., S. J. Mason, and J. W. Fowler. 2006. *Handbook of Production Scheduling*, Chapter Semiconductor Manufacturing Scheduling and Dispatching, 213–241. Boston, MA: Springer US.

Pynnönen, S., and T. Salmi. 1994. “A Report on Least Absolute Deviation Regression with Ordinary Linear Programming”. *Finnish Journal of Business Economics* 43:1:33–49.

Schwaerzel, R. 2006. *Financial Time Series Prediction and Evaluation by Genetic Programming with Trigonometric Functions and High-Order Statistics*. Ph. D. thesis. AAI3255936.

Sha, D., and S. Hsu. 2004, May. “Due-Date Assignment in Wafer Fabrication Using Artificial Neural Networks”. *The International Journal of Advanced Manufacturing Technology* 23 (9): 768–775.

Shanthikumar, J. G., S. Ding, and M. T. Zhang. 2007. “Queueing Theory for Semiconductor Manufacturing Systems: A Survey and Open Problems”. *IEEE Transactions on Automation Science and Engineering* 4 (4): 513–522.

Veeger, C. P. L., L. F. P. Etman, E. Lefeber, I. J. B. F. Adan, J. V. Herk, and J. E. Rooda. 2011. “Predicting Cycle Time Distributions for Integrated Processing Workstations: An Aggregate Modeling Approach”. *IEEE Transactions on Semiconductor Manufacturing* 24:223–236.

Veeger, C. P. L., L. F. P. Etman, J. van Herk, and J. E. Rooda. 2010. “Generating Cycle Time-Throughput Curves Using Effective Process Time Based Aggregate Modeling”. *IEEE Transactions on Semiconductor Manufacturing* 23 (4): 517–526.

AUTHOR BIOGRAPHIES

BIRKAN CAN works in Dell, Limerick, Ireland, as a Senior Advisor in Global Operations Planning. He manages the programs with regards to business demand with Planning leadership and Look-ahead for factories for the EMEA region. Previously, he was a Post-doctoral researcher at the Enterprise Research Centre with a focus on machine learning applications in manufacturing and supply chain industries. He holds a PhD in Operations Research from the same university and BSci in Engineering from Middle East Technical University (Ankara). His research interests include, machine learning and performance improvement in manufacturing systems and supply chains, and machine learning applications to operations research and management. His email address is BirkanCan@gmail.com.

CATHAL HEAVEY is an Associate Professor in the Department of Design & Manufacturing Technology at the University of Limerick. He is an Industrial Engineering graduate of the National University of Ireland (University College Galway) and holds a M. Eng.Sc. and Ph.D. from the same University. He has published in the areas of queuing and simulation modelling. His research interests includes, simulation modelling of

discrete-event systems; modelling and analysis of supply chains and manufacturing systems; process modelling; component-based simulation and decision support systems. His email address is Cathal.Heavey@ul.ie.