Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

TOWARDS SMART MANUFACTURING WITH VIRTUAL FACTORY AND DATA ANALYTICS

Sanjay Jain

Department of Decision Sciences
School of Business
George Washington University
2201 G Street NW, #415
Washington, DC 20052, USA

David Lechevalier

Le2i – Laboratoire Electronique,
Informatique et Image
Université de Bourgogne
Allée Alain Savary
21000 Dijon, FRANCE

Anantha Narayanan

Department of Mechanical Engineering
University of Maryland
College Park, MD 20742, USA

ABSTRACT

Virtual factory models can help improve manufacturing decision making when augmented with data analytics applications. Virtual factory models provide the capability of simulating real factories and generating realistic data streams at the desired level of resolution. Deeper insights can be gained and underlying relationships quantified by channeling the simulation output data to an external analytics tool. This paper describes integration of a virtual factory prototype with a neural network analytics application. The combined capability is used to create a neural network capable of predicting the expected cycle times for a small job shop. The capability can adapt by retraining the neural network whenever the production circumstances change significantly. The trained neural network can be used for functions such as order promising and can support factory management. The analytical and adaptive combination represented by the virtual factory integrated with the neural network thus supports the move towards smart manufacturing.

1 INTRODUCTION

In recent years, interest in smart manufacturing has continued to increase across the globe albeit under different related terminologies. The smart manufacturing term appears to be more prominent in the United States. European Union has its "Factories of the Future" initiative (European Commission 2017) while Germany is actively pursuing its "Industrie 4.0" under the High Tech Strategy 2020 (GTAI 2017). China has its "Made in China 2025" program, India has "Make in India," Korea has "Smart Factory" program, and Japan has "Industrial Value Chain Initiative". There are similar initiatives in many location across the globe. All the initiatives are pursuing use of information and communication technologies for performance improvement and many of them identify modeling and simulation as a key enabler for smart manufacturing. Many of the initiatives also emphasize making the smart manufacturing advancements accessible to small and medium size enterprises (for example see Ezell 2016). Kusiak (2017) recommends

developing platforms that allow collaboration between experts and practitioners for modeling and access to industry data as a major step to enable smart manufacturing.

This paper presents the next step of a development effort with the overall goal of facilitating the move towards smart manufacturing through use of modeling, simulation and data analytics. Jain, Shao, and Shin (2017) discuss the objectives of the effort, describe the virtual factory concept, and present a virtual machine prototype as an initial step towards development of the virtual factory. Jain et al. (2015) describe the implementation of a virtual cell model that utilizes the virtual machine level models to simulate a small cell with turning and milling machines. Jain and Lechevalier (2016) report on the use of standards to facilitate largely automated generation of the virtual cell model as a prototype for such generation of virtual factory level models in future.

This paper reports on utilizing a virtual factory together with data analytics to support the move towards smart manufacturing. The virtual factory is used to model a wide range of production scenarios and collect relevant data. The data is used to train a neural network. The trained neural network can then be used to support the order promising function by rapid generation of estimated cycle times given current status of the production floor. While there have been earlier efforts that use Neural Networks or Data Mining to predict cycle times, they focused on data from the real factory that is limited to scenarios experienced. Use of virtual factory allows generating data for a wide range of anticipated scenarios thus allowing the capability to address situations that may not have occurred previously in the real system. The integrated virtual factory and the neural network application provides the capability to adapt to changing configurations on the production floor. New configurations and/or large changes on the production floor can be adapted to by updating the virtual factory model, regenerating scenarios and using the data to retrain the neural network. The goal is to develop the capability further such that it can be integrated with factory data and control systems to automatically adapt to large scale changes in the system.

2 RELATED WORK

Reported work over the last year in the three relevant areas for this paper are briefly reviewed in this section. Readers are referred to Jain and Lechevalier (2016) for relevant literature prior to last year.

2.1 Virtual Factory

The phrase "virtual factory" is used with multiple definitions in literature. This work uses the definition of virtual factory as a multi-resolution representation across the hierarchical levels of a real factory. More recently phrases such as "Digital Factory" and "Digital Twin" have been used in the literature with similar definitions. Neicu and Savii (2016) utilize a "virtual, digital factory" for testing information flows and evaluating business information systems.

Some efforts emphasize the virtual reality aspects of such representation. Kunz et al. (2016) utilize virtual reality to allow a walk through a virtual factory for planning and evaluation purposes.

2.2 Manufacturing Data Analytics

Reported applications of manufacturing data analytics appear to focus primarily on machine and process level. Chien, Chen, and Wu (2016) use a big data analytics framework to analyze the parameters at process tools level for yield enhancements in semiconductor manufacturing. Mehta, Butkewitsch-Choze, and Seaman (2017) use a data analytics framework for the diagnosis of a process fault in semi-continuous manufacturing. Kache and Seuring (2017) point to lack of empirical research on data analytics at supply chain level and identify opportunities and challenges for such applications based on a Delphi study. Zhong et al. (2017) extend the concept of physical internet concept and big data analytics from supply chain to manufacturing shop floor to identify buffers with high work in progress inventories. Overall, there are limited applications of data analytics at the production floor level in manufacturing.

2.3 Integrated simulation and analytics

Application of simulation and data analytics together for manufacturing is beginning to attract some interest. Tao et al. (2017) utilize digital twins of products and factory together with big data to improve design, manufacturing, and service phases of the product life cycle. They utilize shop floor digital twin primarily for control at the shop floor level. They present a case of drive shaft manufacturing for use of manufacturing digital twin for production task management and for machine level optimization.

Overall, integration of simulation and data analytics offers a good potential for developing decision support systems that can deliver better and faster value than either of these approaches on their own. This work presents an example of the synergy between simulation and data analytics. Such synergy can be harnessed to provide a manufacturing system with the capability to adapt to changing circumstances and thus facilitate the move towards smart manufacturing.

3 APPROACH

3.1 Virtual Factory Prototype

This section presents extensions of the virtual factory prototype proposed by Jain and Lechevalier (2016). The extensions provides improvements for two key components of the virtual factory prototype: (1) the virtual machine model has been extended to more accurately represent the behavior of a machine, and (2) additional data generation capabilities have been developed to facilitate the application of data analytics.

The virtual machine model is modified to include failure as a possible state of the machine. In addition to the failure state, a blocking state is also included to represent a batch held up in a machine if the following machine or buffer area is already full. Figure 1 shows the extension of the state chart in the agent-based model representing the machine.

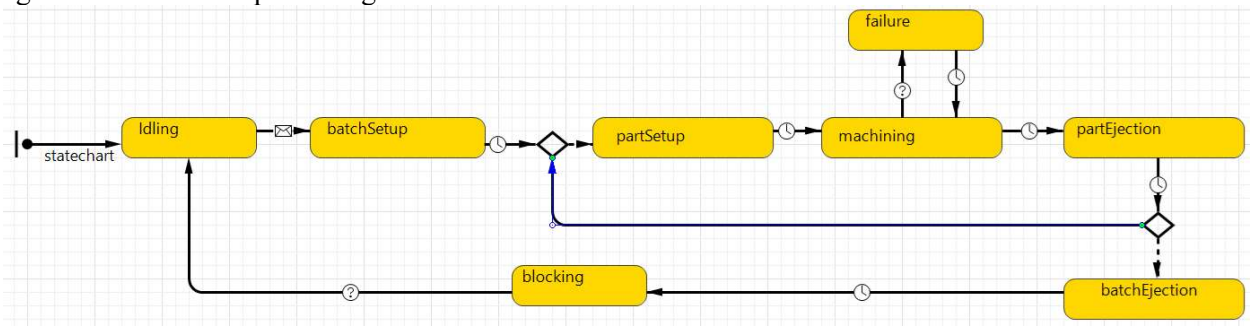

Figure 1: State chart of the machine model.

In the state chart, the *failure* state is included as a conditional state that can occur in the middle of the machining state. The machine goes to the failure state if the total time spent in the machining state is higher than the sampled time to failure. The time to failure follows an exponential distribution with a mean time between failures (MTBF) defined in the Core Manufacturing Simulation Data (CMSD) (SISO 2012) file used as an input of the virtual factory model. The machine model remains in the failure state for a sampled value of repair time. The repair time follows an exponential distribution with a mean time to repair (MTTR) that is also specified via the CMSD file.

The *blocking* state is included after the batch ejection to enable batch holding. Leaving this state depends on the current state of machines or buffer areas that are downstream of the current machine in the process flow. If the current machine cannot send a batch to the following step in the process flow, the batch is held up and the machine remains in the *blocking* state. As soon as the following machine or buffer areas are able to accept a batch, the machine can release the batch. The machine then goes to the idling state and it is open to receive a new batch to process.

Extending the state chart of the virtual machine model provides a more accurate representation of machine behavior. Consequently, a virtual factory model that uses virtual machine models is also more representative of a real factory, and should generate simulation data that are closer to actual data. Simulation data generation capabilities have been also improved by providing features to generate outputs in selected formats including Business To Manufacturing Markup Language (B2MML) which is an XML implementation of the ANSI/ISA-95 (MESA 2010). Two kinds of factory data are represented in B2MML format for each simulation run: (1) number of parts processed for each type produced in the factory, and (2) a list of orders completed.

To summarize the number of parts processed in one simulation run, a B2MML file is created for each type of part possibly produced by the factory. The types of part are originally defined in the CMSD file. Each file contains the following information:

- Type of part
- Duration to produce the number of parts
- Number of produced parts.

Another B2MML file is generated to list the orders that are completed in one simulation run. For each order, the following information are recorded:

- Order ID
- Start time of the order
- End time of the order
- Part type ordered
- Number of parts ordered
- Load of the factory (multiple fields)

The start time is the time when the order is received. The end time is the time when the order is completed. The load of the factory is defined by the work-in-progress (WIP), i.e., the total number of parts that are at various stages of processing in the virtual factory when the order is received. Since different part types might require different times to be processed, the load is represented as a list of numbers that represent the number of parts for each type. For example, if there are two types of parts A and B in a given scenario, the load will contain two numbers to represent the numbers of parts A and the number of parts B that are in WIP inventory when the order is received.

3.2 Neural Network Application

Artificial Neural Networks (NNs) are a computational model used to make predictions from data. An NN is composed of an input layer, zero to several hidden layers and an output layer. Each layer contains at least one neuron. Edges connect the different layers. Weights are assigned to the edges. From a mathematical viewpoint, NNs consist of a set of nonlinear basis functions with free parameters, i.e., weights, that are adjusted. The basis functions are called activation functions. Training the neural network consists of adjusting the weights to minimize the error between the output value of the NN and the real output value for a given data sample. Haykin (1994) provides additional information about NNs.

In this work, we attempt to predict the cycle time for a new incoming job considering the current system load. We build a multilayer-perceptron neural network (NN) to make this prediction. The NN takes certain input parameters, and produces an output prediction. In our study, the input parameters include the type of parts to be manufactured, and estimated current system load. Only a limited set of inputs is used for this proof of concept. A practical application may require consideration of additional factors in the simulation model and the NN such as human operators and supply consistency. The output of the NN is the predicted duration. In our study, we denote the part type by a categorical value limited to a small set of part types. We represent the current system load as a set of numbers, identifying the number

of parts of each part type currently being processed in the machining cells. This serves as a good estimate of the current system load, as it is determined by the amount of time it takes to process each part type. We use stochastic models to represent machine failure and time to repair in the simulation. The effect of these factors is embedded in the generated training data, and thus does not need to be represented as an explicit input to the NN. The power of this approach is that by generating sufficiently large training data through simulations, we can produce a machine learning model that embeds the effects of factors such as failure and variability in resources, thus eliminating the need to model them explicitly as inputs for prediction.

To build the NN model, we must train it with training data from experiments recording the duration to complete orders of various sizes along with the current system load when the orders came in. This data is collected through the simulation described in Section 3.1. We use the simulation to produce a large data set, and partition it into two sets. One set is used to train the NN, and the other set is used to validate the NN. Once the NN has been trained and validated, it can be used to predict the duration for new incoming orders to the factory. When a new order comes in, we give the type and number of parts in the order and the current system load as inputs to the NN. From these, we can obtain the predicted duration for the new order as the output of the NN.

We describe a use case in the next section, with details about the factory model, the data collected, and the prediction model.

4 IMPLEMENTATION WITH A USE CASE

4.1 Use Case

The use case is based on the order promising scenario for a small job shop that produces three part types that are essentially the same part but produced using different materials. Customers place orders for defined quantities of one of the three part types. The planner performing the order promising function has to provide an expected shipment date for the order. The shipment date has to be estimated carefully since a shipment sent too early or too late can lead to a dissatisfied customer and may result in the loss of the customer altogether. It is complex to estimate the shipment date as it can vary based on several factors including the ordered quantity, current load on the shop, and machine failure characteristics. Simulation models that use the current situation on the shop floor as the starting condition can be used for estimating the shipment date, but it can take some time to generate such estimates and may require more expertise than a typical planner may have. Also, the company wouldn't want to have customers wait for several minutes for finding out the shipment date. The planner needs to be supported by an application that generates shipment date estimates for proposed orders within seconds.

The small job shop scenario for the use case is an extension of the virtual machine cell presented by Jain and Lechevalier (2016). The job shop consists of a turning cell with 4 machines and a milling cell with 2 machines. All parts have two operations with first one in the turning cell and the second one in the milling cell. Figure 2 presents the machine shop in this scenario.

For this scenario, Jain and Lechevalier (2016) presented the capability of auto generation of the model including the logic and the layout using a data file based on the CMSD standard. We leverage these capabilities and extended them (as described in section 3.1) to include blocking and machine failures.

In this scenario, a batch is going from a raw material stock to a finished good area after being successively processed in a turning cell, and a milling cell. Each batch is composed of ten parts. Three types of part are defined in this scenario: aluminum, titanium, and steel parts. Each type of part requires different value ranges of machine parameters leading to different processing times and energy consumptions.

Orders are received at the source and specify the part type and the number of parts to be processed in the machine shop. In the base scenario, the order frequency follows a normal distribution with a mean of 60 minutes. During the execution of the simulation, a slider allows the users to modify the frequency.

In the CMSD file used as input, MTBF and MTTR of each machine are specified as below.

- Turning Machine 1: MTBF: 60 hours, MTTR: 2 hours
- Turning Machine 2: MTBF: 40 hours, MTTR: 2 hours
- Turning Machine 3: MTBF: 30 hours, MTTR: 1 hour
- Turning Machine 4: MTBF: 30 hours, MTTR: 1 hour
- Milling Machine 1: MTBF: 50 hours, MTTR: 1 hour
- Milling Machine 2: MTBF: 45 hours, MTTR: 1 hour

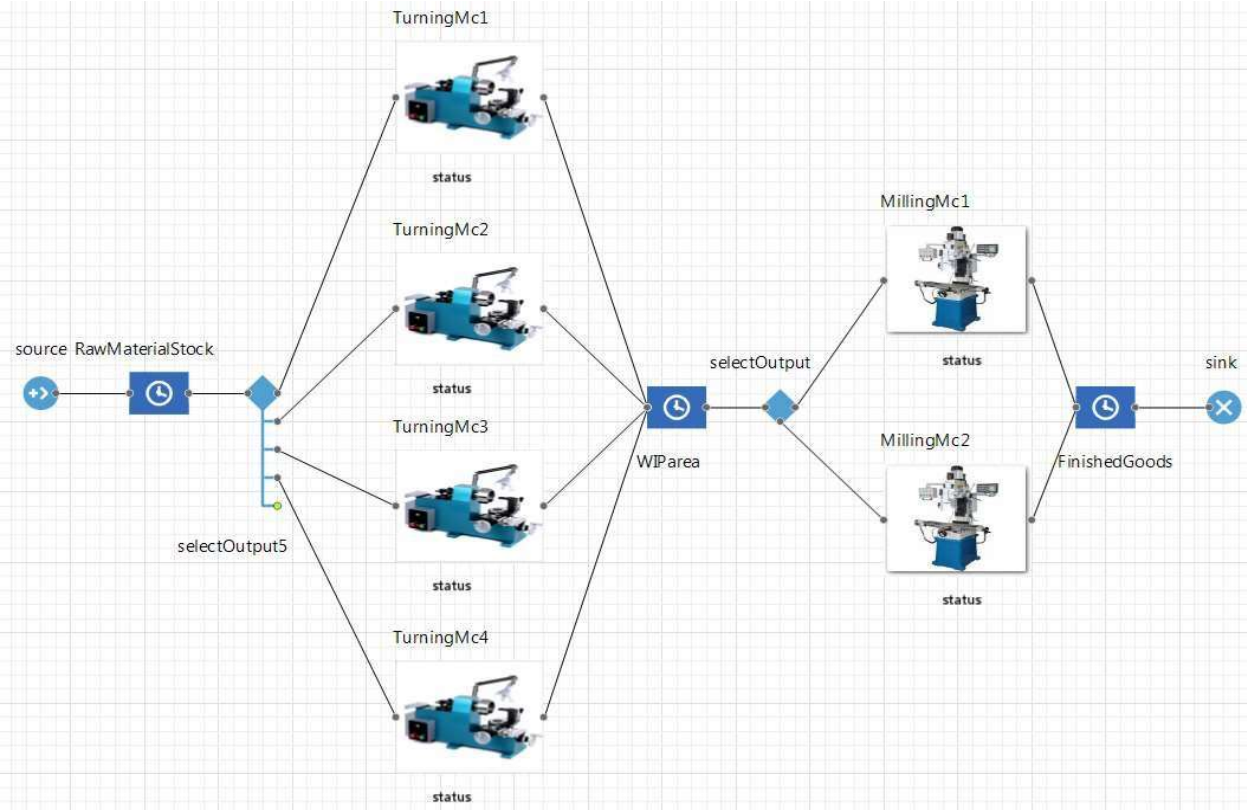

Figure 2: Machine shop scenario (Jain and Lechevalier 2016).

When a machine fails, the machine is unavailable and no batch can be processed on the machine until it is repaired. This obviously impacts the time to process the orders that continue to arrive meanwhile. The next section presents the data generated when this scenario is executed.

4.2 Data generation from Virtual Factory

As mentioned in Section 3.1, the virtual factory prototype has been extended to generate B2MML files that summarizes the orders that have been processed during the simulation. Figure 3 presents an extract of a B2MML file generated when the simulation is over.

For each order, a *ProductionResponse* element is generated and contains the information as described in section 3.1. In this extract, an order "0" of 20 units of "Part1T" was received at 12:00:00AM on March 14, 2017. When the order was received, the machine shop was not processing any part as described with the values "0" in the *Load* element. The order was completed at 1:02:20AM on March 14, 2017.

```
<ProductionPerformance xmlns="http://www.wbf.org/xml/B2MML-V0401">
    <Description>Order summary</Description>
    <ProductionResponse>
        <SegmentResponse>
            <ID>0</ID>
            <ActualStartTime>2017-03-14T00:00:00.000-04:00</ActualStartTime>
            <ActualEndTime>2017-03-14T01:02:20.000-04:00</ActualEndTime>
            <ProductionData>
                <ID>Order</ID>
                <Description>Order</Description>
                <Value>
                    <ValueString>20</ValueString>
                    <DataType>Part1T</DataType>
                    <UnitOfMeasure>integer</UnitOfMeasure>
                </Value>
            </ProductionData>
            <ProductionData>
                <ID>Load</ID>
                <Description>Load</Description>
                <Value>
                    <ValueString>0</ValueString>
                    <DataType>Part1A</DataType>
                    <UnitOfMeasure>integer</UnitOfMeasure>
                </Value>
                <Value>
                    <ValueString>0</ValueString>
                    <DataType>Part1S</DataType>
                    <UnitOfMeasure>integer</UnitOfMeasure>
                </Value>
                <Value>
                    <ValueString>0</ValueString>
                    <DataType>Part1T</DataType>
                    <UnitOfMeasure>integer</UnitOfMeasure>
                </Value>
            </ProductionData>
        </SegmentResponse>
    </ProductionResponse>
```

Figure 3: B2MML file to summarize the order processing.

Two different B2MML files have been generated. One file summarizes the orders processed during a simulation without failures of machines. The other file summarizes the orders processed during a simulation with failures of machines. To facilitate the data manipulation by an analytical model, we also developed a converter of B2MML files into CSV files. Figure 4 shows the extract from CSV file corresponding to the B2MML extract shown in Figure 3. The CSV file contains the order ID, the description of the order, the duration to process the data in seconds, the number of parts ordered, the type or parts ordered, and the current system load represented by the number of parts in process. Two CSV files have been generated from the two different B2MML files generated. These two CSV files are used to train neural networks for the two scenarios. This training is described in the next section.

```
ID,Description,Duration,Number,Type,Part1A,Part1S,Part1T
0,Order,3740,20,Part1T,0,0,0
```

Figure 4: CSV file to summarize the order.

4.3 Neural Network Training

The NN model tries to make a prediction on a value of interest, based on our current knowledge about other system variables that are believed to have an impact on the value of interest. In our case, we are interested in predicting the duration to complete a new order. The known information includes the order itself, and the current system load. The order information is represented using a categorical variable to specify the material type, and a numerical variable to specify the number of parts in the order. The load on the system is represented by a set of numerical values, one for each part type currently being processed.

We used the Knime data analytics platform to train and validate the NN model. The data generated by the virtual factory simulation in Section 4.2 is used for training and validation. We used Knime's MultiLayerPerceptron (MLP) model to build the NN. Knime provides an MLP Learner node that employs a reverse propagation algorithm to train a multilayer perceptron NN from tabular input data. The Knime MLP learner only accepts numerical input values. One of our input variables, "part type", is categorical. In order to use this trainer, we transform the categorical "part type" variable into three separate columns, one for each part type. The "part type" value is then transformed into a value of 1 for the column corresponding to the one in the order, and 0 for the other columns. The NN was configured with a single hidden layer with 10 hidden neurons. The "Duration" column was marked as the target for prediction.

The data collected in Section 4.2 was partitioned into two sets, one for training and one for validation. The training data was then normalized and sent to the MLP Learner node, which produced a trained NN model. We conducted two simulation experiments. The first simulation only modeled load on the system in terms of the number of parts of each type being processed, and did not include any failure. The second simulation included failures, modeled as described in the previous sections. We obtained 2,264 rows of data for the first scenario, and 1,720 rows of data for the second. The data sets were partitioned into 80% for training, and 20% for validation. We built separate neural networks for the two experiments. We provide the results of the validation below.

4.4 Neural Network Validation

Validation for a NN is defined as testing its ability to predict values for the variable of interest close to the actual values realized in the system. The NN generated by the MLP Learner node is passed to a predictor node, where it is used to predict the duration (cycle time) value for the validation portion of the data. The validation portion of the data collected in Section 4.2 is normalized and passed to the predictor to obtain the predicted duration. The prediction is denormalized, and compared with the original duration column in the validation data. Figure 5 shows the graph of the predicted and original values of the duration for a subset of the validation data set, for the first scenario without failures. The Y-axis on the graph shows the duration in seconds, and the X-axis is the series of batches of orders. Since the validation partition was randomly sampled from the dataset, these points do not necessarily represent a continuous sequence of incoming orders. However, a high duration value should be associated with a high system load at the time the order arrived. The validation had an R^2 value of 0.998, and mean absolute error of 1,200 seconds. The average duration in the validation data set was 54,371 seconds. This represents an error of about 2%. The RMS error was 2,311 seconds.

Figure 6 shows the graph of the predicted and original values of the duration for the scenario with failures. In this case, a high value of duration is associated with a high load on the system, and/or possible machine failures. For this scenario, the validation had an R^2 value of 0.996, and mean absolute error of

1,716 seconds. The average duration in the validation data set was 57,201 seconds. This represents an error of about 3%. The RMS error was 2,463 seconds.

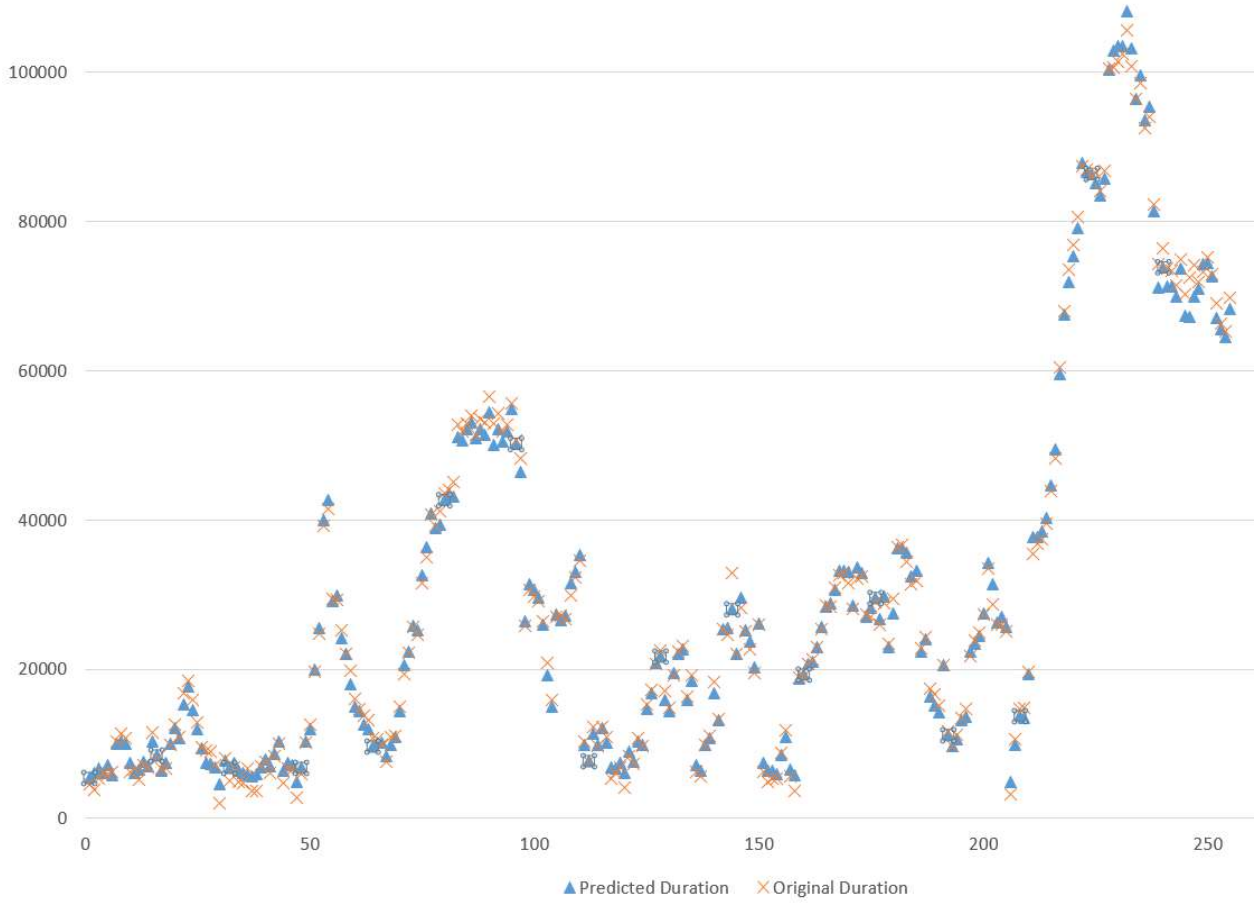

Figure 5: Predicted vs. original durations for no machine failure case.

The validation results show that in both scenarios, we are able to get a good predictive model from the simulated data. We notice that the prediction is less accurate but still within acceptable limits when the load on the system is high. The prediction accuracy only drops slightly when we consider the case with machine failures. The results show that representing system load and machine failure as described in this paper provide a good basis for building a prediction model for these types of systems.

4.5 Discussion

The NN validation results show that it provides quite accurate predictions of the flow time of part batches through the small job shop considered in the use case. Once the NN has been built and trained, it can provide the flow time predictions instantly, and thus provide the quick response needed for order promising functions by the planners. The prediction is within 3% of the actual durations in our tests.

In our approach, we take advantage of simulations to generate large amounts of data, which allows us to simplify the representation of factors like current system load and variabilities due to machine failure and downtime. We represent load as the number of parts of each part type currently being processed in the machining cells. We use a stochastic model for machine failure and downtime, and the effect of machine downtime is embedded in the generated training data, and does not have to be modeled explicitly. An alternative approach can be to represent machine failure as an additional load attribute, for

which training data can be generated using stochastic models. In the latter approach, the prediction can take into account a manually specified value for machine conditions based on the current factory conditions to improve the accuracy of predictions. This can be extended to other factors beyond machine failure such as resource variability.

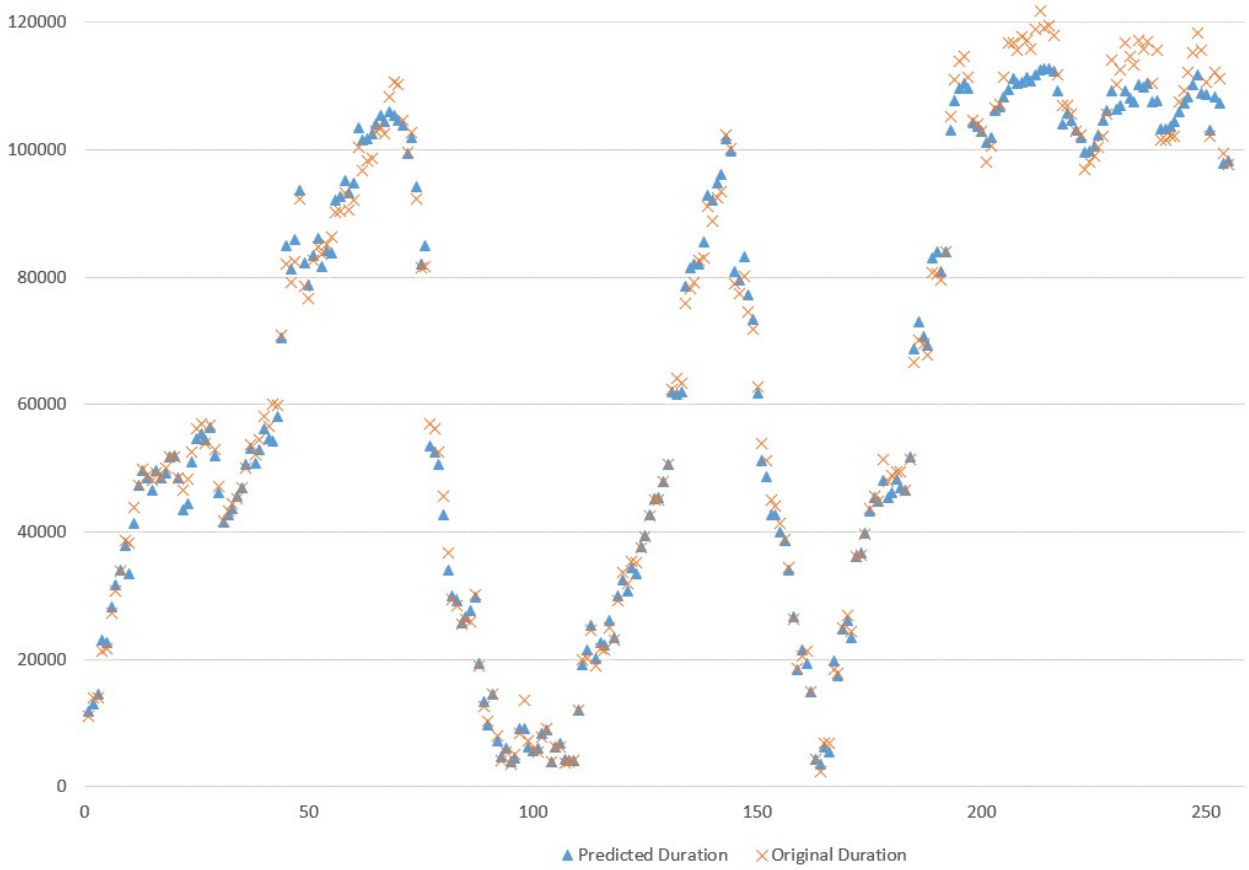

Figure 6: Predicted vs. original durations for the case with machine failures.

The NN has been trained for the defined configuration and for the range of load levels included in the simulation runs conducted using the virtual factory prototype. If there are significant changes in major parameters such as number of machines, number of part types, process plans, machine failure characteristics, etc., the NN will need to be trained again. This will require updating the input files to reflect the current real factory, regenerating the corresponding virtual factory, and running through a range of load level scenarios to generate data to train and validate the NN. Such significant changes are expected to happen only a few times or less in a year and hence are not considered as a deterrent for application of the presented approach in real systems. The use case utilized a very small job shop scenario. The capability of virtual factory and NN combination to predict flow times in larger systems will need to be tested for evaluating its applicability.

5 CONCLUSION

Global competition is pushing industry in the developed world to advance their manufacturing systems through pervasive use of analytics. The goal is to make the systems smart to allow them to autonomously monitor and control various aspects to achieve increasingly efficient and effective operations. A number of initiatives are in progress across the globe towards this goal. This paper describes a small step towards

one of the many aspects of this goal via the combination of simulation and analytics to predict flow times of incoming orders. The simulation component is implemented as a virtual factory prototype that allows largely auto-generation of a simulation model based on data for a limited set of manufacturing configurations. The analytics component is implemented as a largely auto-generated neural network. The two components have been integrated using information flows that are based on standard formats where available. The cycle time predictions by the integrated virtual factory and neural network capability are based on the current load level of the manufacturing system and thus adapt to operational variations in the system. The combined ability can also adapt to significant changes in the system through updating the neural network based on simulations of the updated system.

The presented application was shown to be capable of providing quite accurate predictions of flow time for the order promising use case for the considered small job shop scenario. Future directions under consideration include enhancing the virtual factory prototype's modeling capabilities to include such factors as human operators, different material handling devices, and other realistic constraints, testing of the combined simulation-analytics approach for scenarios with larger manufacturing systems, and generating standard data sets for use as benchmarks for manufacturing data analytics applications.

DISCLAIMER

No approval or endorsement of any commercial product by the National Institute of Standards and Technology (NIST) is intended or implied. Certain commercial software systems are identified in this paper to facilitate understanding. Such identification does not imply that these software systems are necessarily the best available for the purpose.

ACKNOWLEDGMENTS

The work of David Lechevalier and Anantha Narayanan on this effort was supported by National Institute of Standards and Technology's Guest Researcher Program and under Cooperative Agreement No. 70NANB14H250 respectively.

REFERENCES

Chien, C.-F., Y.-J. Chen, and J.-Z. Wu. 2016. "Big Data Analytics for Modeling WAT Parameter Variation Induced by Process Tool in Semiconductor Manufacturing and Empirical Study". In *Proceedings of 2016 Winter Simulation Conference*, edited by T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick, 2512-2522. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

European Commission. 2017. "Factories of the Future". http://ec.europa.eu.proxygw.wrlc.org/research/industrial_technologies/factories-of-the-future_en.html.

Ezell, S. J. 2016. "A Policymaker's Guide to Smart Manufacturing". Information & Technology Innovation Foundation. https://itif.org/publications/2016/11/30/policymakers-guide-smart-manufacturing.

GAIT. 2017. "Industrie 4.0". Germany Trade & Invest. https://industrie4.0.gtai.de/INDUSTRIE40/Navigation/EN/Topics/industrie-4-0.html.

Haykin, S. 1994. "Neural Networks: A Comprehensive Foundation". Macmillan, NY, 1994.

Jain, S., D. Lechevalier, J. Woo, and S.-J. Shin. 2015. "Towards a Virtual Factory Prototype". In *Proceedings of 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, 2207-2218. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Jain, S., and D. Lechevalier. 2016. "Standards Based Generation of a Virtual Factory Model". In *Proceedings of 2016 Winter Simulation Conference*, edited by T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick, 2762-2773. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Jain, S., G. Shao, and S.-J. Shin. 2017. "Manufacturing Data Analytics Using a Virtual Factory Representation". *International Journal of Production Research* 55(18): 5450-5464.

Kache, F., and S. Seuring. 2017. "Challenges and Opportunities of Digital Information at the Intersection of Big Data Analytics and Supply Chain Management". *International Journal of Operations & Production Management* 37(1): 10-36.

Kunz, A., M. Zank, M. Fjeld, and T. Nescher. 2016. "Real Walking in Virtual Environments for Factory Planning and Evaluation". *Procedia CIRP* 44 (2016): 257-262.

Kusiak, A. 2017. "Smart Manufacturing Must Embrace Big Data". *Nature* 544(7648): 23.

Mehta, P., S. Butkewitsch-Choze, and C. Seaman. 2017. "Data Analytics Framework for Semi-Continuous Manufacturing Process—Implementation Vision with a Use Case". *Journal of Manufacturing Systems,* in press, https://doi.org/10.1016/j.jmsy.2017.04.014.

MESA. 2010. *Building a Manufacturing Transformation Strategy with ISA-95 Methods.* White paper #38. Chandler, AZ, USA: MESA International.

Neicu, M.Ş., and G.G. Savii. 2016. "Evaluation of Algorithms and Methods for Developing Business Information Systems Using Virtual Factory". In *Applied Mechanics and Materials*, 841:367-372. Trans Tech Publications, 2016.

SISO 2012. *SISO-STD-008-01-2012: Standard for Core Manufacturing Simulation Data – XML Representation.* Simulation Interoperability Standards Organization, Orlando, FL.

Tao, F., J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui. 2017. "Digital Twin-Driven Product Design, Manufacturing and Service with Big Data". *The International Journal of Advanced Manufacturing Technology*, 1-14. DOI: 10.1007/s00170-017-0233-1.

Zhong, R.Y., C. Xu, C. Chen, and G.Q. Huang. 2017. "Big Data Analytics for Physical Internet-based intelligent manufacturing shop floors". *International Journal of Production Research* 55(9): 2610-2621.

AUTHOR BIOGRAPHIES

SANJAY JAIN is an Associate Industry Professor in the Department of Decision Sciences, School of Business at the George Washington University. Before moving to academia, he accumulated over a dozen years of industrial R&D and consulting experience working at Accenture in Reston, VA, USA, Singapore Institute of Manufacturing Technology, Singapore and General Motors North American Operations Technical Center in Warren, MI, USA. His research interests are in application of modeling and simulation of complex scenarios including smart manufacturing systems and project management. His email address is jain@email.gwu.edu.

DAVID LECHEVALIER is an international guest associate in NIST's Systems Integration Division of the Engineering Laboratory. He completed his PhD at the University of Burgundy in 2017. His current research topics include advanced analytics and modeling for Smart Manufacturing. He received a computer science Master's degree in Database and Artificial Intelligence from University of Burgundy in 2012. His email address is david_lechevalier@etu.u-bourgogne.fr.

ANANTHA NARAYANAN is a Research Associate at the University of Maryland. He completed his PhD at Vanderbilt University, Nashville, TN, USA in 2008. His research interests are in data analytics and model based systems engineering. He is currently working as a guest associate in NIST's Systems Integration Division of the Engineering Laboratory. His email address is anantha@umd.edu.