Predictive AI To Feed Simulation

Carlo Simon, Stefan Haag and Natan Georgievic Badurasvili
Hochschule Worms
Erenburgerstr. 19, 67549 Worms, Germany
e-mail: {simon, haag, natan.badurasvili}@hs-worms.de

***Abstract*—In an industry project, the authors had successfully modelled and simulated the inbound and outbound traffic of a warehouse with the aid of high-level Petri nets. But instead of taking this simulation tool to improve the future planning, the practitioners confronted the authors with another problem: the reasons for the incorrect planning is less the planning process but the inability to foresee which of the scheduled trucks will be late and sabotage the time plan. As a solution to this problem, the authors considered methods of predictive artificial intelligence and especially machine learning. The idea is to take past schedules to train a neural network in order to forecast deviations of upcoming schedules. The paper answers the question on how to extend the previous simulation model by a suitable forecast component which is now ready to be tested with real-world data. The paper explains the scenario of the real-world warehouse, the simulation of its traffic and the information needed for this. Afterwards, the necessary extensions of the data set are explained and how to set up a machine learning component to predict future deviations of schedules. The adapted schedules can then be simulated to create alternative schedules. This is a next step of the authors' research to conduct their simulation on a set of future scenarios in order to chose the schedule that performs not worse than the initial schedule on the original data, but performs better under the alternative scenarios.**

***Keywords*-Predictive Artificial Intelligence; Neural Networks; Machine Learning; Simulation; Petri Nets.**

I. Introduction

During the SIMUL 2023 conference, the participants had various discussions about the impact of currently discussed Artificial Intelligence (AI) methods like transformers on simulation. While simulation is about causality, many AI methods are about correlation. The participants generally doubted that AI will substitute current simulation methods. However, there is still the possibility to pair both approaches.

For the authors, this discussion happened at the right time, because at this moment they were incapable to solve a problem that arose in an industry project with their currently preferred methods. They could simulate the incoming and outgoing traffic of a large warehouse for registered transports as described in Section II with the aid of a high-level Petri net. But this did not take into account that the registered (planned) arrival time and the actual arrival time of the transports often do not match.

The authors started to identify possible reasons for late transports, such as the transport distance, the shipping agent, the producer or the current weather conditions. All of them did not play a role during the actual simulation of the inbound and outbound traffic of the warehouse.

Unfortunately, the industry partner does not collect information about delays and, hence, cannot provide empirical values. Before asking them to do so, it was the wish of the authors to develop their simulation environment further by adding AI technologies, first. Two possible approaches have been considered initially:

1) Develop an AI that applies compensation strategies comparable to those of the warehouse management if arriving transports are late.
2) Develop an AI that forecasts the expected arrival time of transports on the base of former transportation data.

The first approach has been discarded because this would require a close observation of the current work. The authors did not expect to receive an approval for this.

The second approach, however, seemed to be a natural evolution of work conducted up to now. The idea led to the research question:

> *How can we extend the previous simulation model by a forecast component based on machine learning?*

In this phase of the research, it is not yet the aim to apply the approach to real-world data. It is rather the aim to show the feasibility of the approach, and to develop a framework as a foundation from which concrete industry projects can be processed.

Before explaining the new AI specific components of the simulation environment, the paper continues in Section II with a description of the high-level Petri net simulation model for the inbound and outbound traffic of a warehouse. The authors especially focus on the data needed for simulation. Afterwards, in Section III, a brief introduction is given to the necessary Machine Learning (ML) methods. Both topics are combined in Section IV to define the structure of a possible ML data set for the problem first and to explain the ML approach in Section V next. Section VI demonstrates how to integrate the ML approach into the existing environment. The paper ends in Section VII with a conclusion and an outlook on future work.

II. Real-World Problem

The industry partner at an industrial park (Industriepark Höchst, *ISL*) provides logistics services for chemical, pharmaceutical and healthcare industries and currently expands its logistics services. Freely published information show the size of this venture [1]:

- Space for more than 21,000 pallets
- 9 separate warehouse sections
- Storage of multiple hazardous material storage classes for chemicals and pharmaceutics
- A wide temperature range from -6 to 20 degrees Celsius in the different sections of the warehouse

As depicted in Figure 1, the warehouse can be accessed via ramps (2). Each section has a loading zone (3) and the actual storage zone (6).

During planning and go live, the inbound and outbound traffic of this warehouse has to be simulated to objectify assumptions made during the conceptual phase. A typical inbound process is conducted as follows: before approaching the industrial park, the shipping agent books a time slot in advance. When trucks arrive, the drivers register with the gatekeeper in the office (1). Afterwards, they dock at the ramp they have been assigned to (2). Then, the goods are picked by standard forklifts (called VHS) and placed in the staging zone (3). After the truck has left, the goods are carried through the driving zone (4) and dropped on a handover point (5). Finally, narrow aisle forklifts (called SGS) pick the goods and store them in the high rack storage area (6).

Outbound processes are executed in reverse order, except that the goods are provided in the staging zone before trucks arrive.

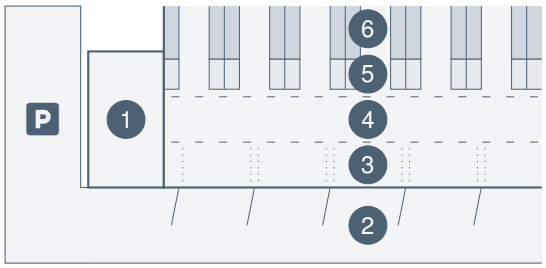

Figure 1. Sketch of the Warehouse.

Various specifics may be excluded from the simulation, such as the commissioning of certain goods. Furthermore, the precise positioning of goods within the warehouse, and consequently the exact driving times, hold diminished importance. In lieu of this, reasonable average times have been selected to replicate typical operational norms.

Model and simulation have been described in detail in [2] and [3]. The model consists of state machines for the inbound and outbound processes and a high-level Petri net to execute these state machines in parallel. Also, time constraints and restricting resources are taken into account.

Two conceptual data models for orders and resources are needed for this. The resources are discussed, first. Table I shows the data needed for simulation based on one kind of resource. This information is spread over several tables (or, in terms of Petri nets, over several places) for each kind of resource.

TABLE I. ATTRIBUTES FOR THE RESOURCE PLACES

Attribute	Description
id	id for resources of this kind
product	product group
free	available or locked
timestamp	timestamp of the latest state change
order	assigned order id

Attribute *id* may identify a specific resource like a numbered ramp or a dedicated resource of this kind, like one of the VHS. *product* describes the resource allocation to chemical or pharmaceutical. *order* references to the order that uses this resource and simplifies joins among the different data sets.

Table II shows the data needed for orders. Beside an identifier *id* and the specification whether the order belongs to a chemical or pharmaceutical *product*, the *total* number of pallets for the transport is specified. *status* identifies the current order's state and, implicitly, whether this is an inbound or outbound process.

TABLE II. ATTRIBUTES OF AN ORDER'S STATE

Attribute	Description
id	order id
product	product group
total	total amount of pallets requested
status	initial or current order status
ramp	target ramp
arrival	scheduled time of arrival
preparation	scheduled time of completed staging
fillHandover	amount of pallets in handover areas
fillRamp	amount of pallets at target ramp
fillTruck	amount of pallets in truck
usedGate	used resource *gate*
usedSGS	used resource *SGS*
usedVHS	used resource *VHS*
timestamp	timestamp of the latest state change

One or two times must be defined per order: for both inbound and outbound, the *arrival* time of the truck is given due to its registration. Outbound processes additionally need a *preparation* time when staging begins. This staging time leaves room for optimisation for the warehouse operators.

At the end of the simulation explained in the following, all changes to orders are exported to a dashboard. A *timestamp* traces the moments these changes occur. To simplify the computing of this visualisation, the allocated resources like *ramp*, *usedGate*, *usedSGS*, *usedVHS* are saved. Finally, the amount of goods at the different places is stored in the attributes *fillHandover*, *fillRamp*, and *fillTruck*.

Without having an impact on the result, the real process and the simulated one differ slightly. For the simulation, the ramps are assigned in advance; in real world, the ramp is assigned by the gatekeeper.

The authors have chosen the *Process-Simulations.Center (P-S.C)* for modelling and simulation [4]. Since the *P-S.C* is a Petri net based Integrated Management System, it fulfils further constraints important for the industry partner among the pure ability to simulate. Safety and security aspects are of high priority to ISL. Therefore, access to and visibility of (real-world) data must be limited.

The *P-S.C-Cloud* (the *P-S.C* is only provided via internet) and the multi-client capability of the *P-S.C* help to manage security issues [4]: The tool itself only runs locally in a browser with data never leaving the system. Sensitive input data and simulation results can be stored on in-house servers without the *P-S.C-Cloud* ever coming in contact.

In addition, the industry partner requires a user interface (UI) to edit the simulation parameters (orders, times, resources,

priorities) easily. The simulation results ought to be presented in a descriptive dashboard. Today, such an interface is called a digital shadow, a piece of software which maps real-world data and processes to a virtual world [5].

Figure 2 explains the interaction between the *P-S.C* and the local UI using CSV-files. Simulation parameters can be imported this way, too.

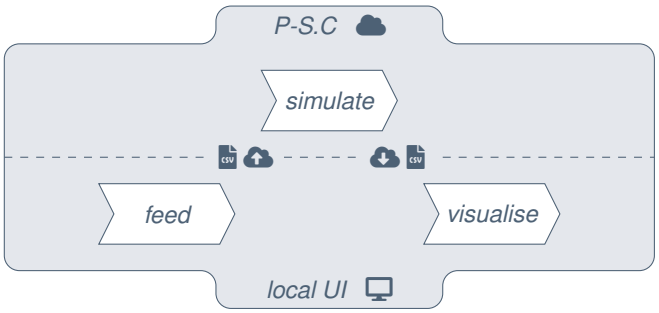

Figure 2. Coupling of Digital Shadow (local UI) and *P-S.C*.

The execution of a specific simulation scenario is divided into three distinct phases:

- **Feed:** The local UI is implemented as a web-based application that can be used with any internet connected computer or iPad. It is used to enter the simulation parameter, priorities and especially the actual order data.
- **Simulate:** The *P-S.C* loads the data into the browser of the end user to start the simulation. The *P-S.C-Cloud* does not get in touch with it which guarantees full control over the data. After the simulation has finished, an automatic download is started and the users can store the results on local hardware.
- **Visualise:** The downloaded file in turn can now be uploaded in a dashboard which is also implemented in the local UI. This helps the end-users to find the best strategy for driving the warehouse. In particular, workload spikes, transportation bottlenecks, and staff occupancy can be analysed and visualised.

This approach is limited to simulate one specific schedule of orders for one specific period, e.g., one day. It is not flexible concerning deviations of the transports. The next sections explain how to extend this environment to handle this limitation.

III. Machine Learning Fundamentals

Artificial Intelligence (AI) is defined in many different ways. These definitions lie in a quadrant chart: one axis denotes the scope of acting and thinking, while the other denotes the spectrum between pure rationality and human semirationality. The most common definition of AI is that of a rational agent. Such an agent tries to provide the best (expected) outcome, given its inputs. What constitutes the best outcome remains a matter of definition [6].

If a rational agent is created to improve the provided output by gaining some sort of experience, this is called Machine Learning (ML). Figure 3 shows the idea of ML as described by [7]. The figure depicts the rational agent as a model. Its task (red) is to compute an output when presented with input data. The model obtains a mapping based on training data. Its formation is the learning problem (blue). Which algorithms are used depends on the choice of ML model [7].

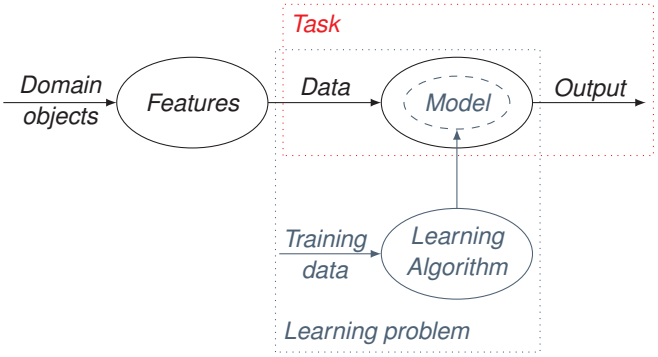

Figure 3. Machine Learning as Explained by [7].

One possible ML model is Deep Learning (DL). A DL model is an artificial neural network with several (hidden) layers. A neural network is to mimic the principle of real neurons. Single neurons are connected by directed, weighted arcs. If the incoming arcs yield a high enough activation potential, the activation function in a neuron triggers a corresponding output. The weighs constitute the main parameters of this ML model type [8].

If the training set includes results, this is called supervised learning. In this case, the data is considered labelled. The metric used to examine the model's quality is to compare the actual results with the computed ones [9].

Two kinds of problems may be solved with the aid of artificial neural networks:

- **Classification:** In mathematical notation, a classifier is a function $y = f(x)$, where x is the input data item and y is the output category [10].
  Applied to our example of a warehouse, classification could help to predict which transports may be unpunctual.
- **Regression:** In regression, we try to understand the data points by discovering the curve that might have generated them [10].
  Applied to our example of a warehouse, regression could help to predict how many minutes transports may be unpunctual.

Both aspects will be discussed in Section V.

IV. Extended Data Set for ML

The aim of the presented approach is to complete future orders and especially to forecast

- which future transports will or will not be in time, i.e., to solve a classification problem, and
- to forecast the expected deviation of the arrival time of future transports, i.e., to solve a regression problem.

But which parameters may influence the arrival time of trucks? And how can this information be coded, such that it can be processed by a learning algorithm?

First, it is worth to consider the data used for simulation already. All "internal" parameters of the orders like order *id*, current *timestamp* of the simulation, allocated *ramp* and other resources (*fillHandover*, *fillRamp*, *fillTruck*, *usedGate*, *usedSGS*, or *usedVHS*), and the *preparation* time for outgoing orders can be ruled out as possible sources.

The probably most valuable source is the planned *arrival* time coded as a timestamp consisting of date and time. This information, which is typically stored as a string, should be preprocessed for a learning algorithm. The following information can be extracted and coded as discrete numbers:

- The month or season of arrival, which may have an impact due to weather conditions.
- The arrival time in hours or at least in categories like early, in the middle of the day or by the end of the day, which may have an impact on the transportation conditions like traffic or the work schedule of the drivers.
- The day of week which may have an impact on the traffic intensity.

Also, the *state* attribute may be of interest because it is used to differ between incoming and outgoing transports. For outgoing transports, "only" an empty truck has to arrive while incoming transports need to be prepared before they reach the warehouse. This might have an impact on the arrival time.

Attribute *product* is not as valuable as supposed by its name. The simulation only distinguishes between chemical and pharmaceutical products which is very rough. It is not obvious that these two categories correlate with a deviation since there are hundreds of different concrete products that belong to these two groups.

Finally, the *total* amount of pallets is a candidate which might have an impact on the arrival time, especially for incoming transports. Large amounts of goods may cause more problems when being loaded compared to smaller ones.

Further attributes might have an impact which are not considered during simulation. The following attributes have been found by the authors:

- The transportation *distance*, especially if a truck has to arrive from outside of the industrial park or whether it is an internal transport.
- The *shipping agents* may differ concerning their quality standards and the accuracy of delivery time.
- Finally, the producer might have an impact on the time a transport can start after being loaded and, hence, whether it arrives on time.

All remaining values can be enumerated and can hence be used for training of a neuronal net.

Unfortunately, the industry partner does not track these additional information. Even the deviation between the planned and the actual arrival time is not documented. Hence, the following considerations are only of theoretical nature.

V. ML FOR PREDICTIVE DEVIATION SCHEDULING

In the context discussed in this paper, Deep Neural Networks (DNNs) can be used threefold. First, they can create yet missing data, thus establishing a means to plan ahead of knowledge. Second, they can predict which transports may be not on time. Third, they can surmise the time deviation.

A. Neural Nets to Complete the Feed

Data becomes worse - or even non-existent - the farther the look into the future. Thus, missing but plausible data has to be integrated into the feed. The corresponding DNN gets trained with historical data of planned arrival times of trucks. From this training data, the net creates fictional yet plausible data entries to complete the fragmentary known data. The thus enriched data constitutes one possible scenario to be analysed by simulation. As it is the first fully scheduled data set, it can be regarded as the base scenario.

B. Neural Nets for Classification

The associated DNN can learn from historical data which deliveries and dispatches were on time. Thus, it can predict the probability of a trip to be delayed or advanced. This is due to the neural nets capability to learn from underlying correlations. Such correlations may be interpreted as questions like:

- Are there shipping agents that often are late?
- Are there producers that always are early?
- Are midweek deliveries more reliable than ones on Mondays?

After establishing the probabilities of unpunctuality, alternative scenarios can be established. These scenarios incorporate differing yet still plausible delay and advance times. The corresponding schedules can then be compared to the base scenario.

C. Neural Nets for Regression

Knowing which delivery may be late is one side of the coin. The other is the actual time. The fitting DNN can make guesses about these times. Again, the capabilities of neural nets to represent correlations prove useful: Several different effects may overlap that may add up to massive delays. An example for this may be an unreliable hauler in the midst of winter on an early Monday morning. These time delta represent the last puzzle piece in creating alternative scenarios.

D. Implementation Insights

As a tool only provided via internet, the architecture of the *P-S.C-Cloud* consists of a JavaScript client and a PHP Server. The prototypical implementation of a ML component to conduct the previously described tasks is done with Python instead. The reason for this is the existence of a large number of ML libraries that can simply be combined. Especially the following packages are used [11]:

- **NumPy** provides data types and functions for easier handling of complex structures, such as vectors and matrices.

- **pandas** is designed for more complex structures and their easy handling. One strength is its extensive functionality for table structures.
- **Matplotlib** is used for visual analyses and plotting.
- **scikit-learn** contains many ML algorithms that can be easily used in your own program.
- **Keras** can build artificial neural networks.
- **TensorFlow** extends Keras with additional well performing functionalities and can handle large and complex data structures.

With the goal to train a DNN as an example, typical delivery information including possible delays, have been guessed in a students' project. While some of the information have been randomised, others include pattern like specific shipping agents always being late.

The numerical representation of the input data is partially generated in the JavaScript part of the implementation, others make use of the predefined ML algorithms in Python.

The derived information concerning completed order lists, classification and regression are currently produced in the Python environment. They can be stored in CSV files which can then be integrated into the *P-S.C-Cloud* via file upload.

VI. ML Extended Simulation

It is the aim to integrate the ML solution presented in the previous section into the formerly introduced simulation environment as shown in Figure 4. After the trusted orders are entered for simulation, a new phase **refurbish** is executed which makes use of the ML model.

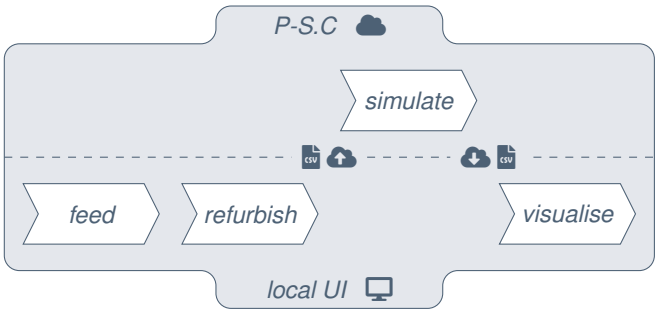

Figure 4. Simulation environment extended by ML.

With the goal to improve the ramp allocation, simulation is conducted in two phases. Figure 5 shows the improved planning approach.

A. Simulation of a Planned Schedule

In the first iteration, the orders are simulated for the case that all transports arrive as planned. If further orders are expected for the simulation period, the ML algorithm is able to generate plausible orders with respect to the known order history.

The simulation is conducted with the goal to find an optimal ramp allocation which reduces idle times for the shipping companies combined with a minimal labour utilisation. Different ramp allocations may lead to almost same results.

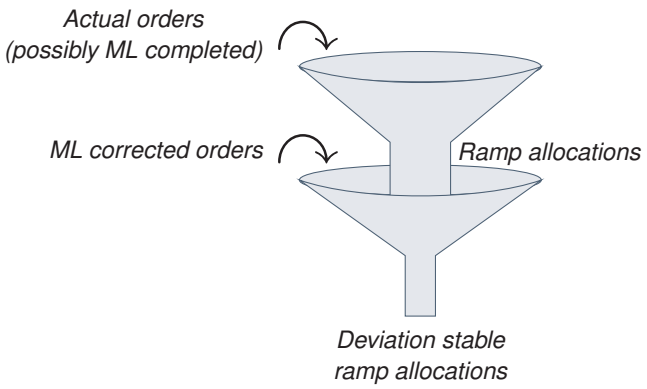

Figure 5. Two-Phase approach for ramp allocation.

B. Simulation of a Planned Schedule plus (DNN) Delays

During the second phase, the orders are corrected with respect to the assumed deviations of transport times. The ramp allocation strategies found in the first phase are applied to these adapted schedules and evaluated with respect to the optimisation criteria. These simulation results are used as a filter to find that allocation strategy which fits best to the original and the adapted orders.

C. Preliminary Work of Industry Partners

As mentioned before, the industry partner does not collect information needed for the machine learning approach. To prepare data collection, the training data mentioned before is divided into three classes:

- **Available:** timestamp of the planned arrival; incoming vs. outgoing transports; amount of pallets
- **Not available but collectable:** the delay of transports; transportation distance in the sense of transports inside the industry park or nor; the shipping agent; the producer
- **Not available and not collectable:** in advance, the transportation distance in length

This classification reinforces the appraisal that the introduced approach can be conducted. It's application depends on the willingness of the industry partners.

VII. Conclusion and Outlook

As initially stated, the goal of this research is to provide an environment to improve simulation in logistics. It uses machine learning based techniques to create schedules that are (mostly) indifferent to deviations from the original planning. Thus, a good schedule performs at least as good on the original data as the original schedule; but it performs as good as possible under differing scenarios.

A prototypical environment has been implemented, but the parts are not fully integrated yet. The reason for this is that the industry partner does not collect information of the deviation of planned and actual arriving time of the trucks. Also, other information that could help to train a neural net are not

considered by the partner. The possibility to do so, has been explained above.

This is the next important research step for the authors: check the approach with real-world data! Afterwards it makes sense to develop the integration further.

Even with this approach it is not possible to know which transports will be late in the future. But the developed scenario is more stable with respect to delays in delivery without degrading the initial plan. And for checking this plan, simulation is still needed. Statistical estimations are insufficient for processes in logistics.

REFERENCES

[1] Infraserv Logistics GmbH, *Overview hazardous substances warehouse*, https://www.infraserv-logistics.com/en/isl/news/news/ (last accessed 08.2024), 2023.

[2] L. Zakfeld, C. Simon, S. Hladik, and S. Haag, "Real World Case Study To Teach Simulation", in *SIMUL 2023: The Fifteenth International Conference on Advances in System Simulation*, Valencia (Spain), 2023, pp. 19–24.

[3] C. Simon and S. Haag, "Pairing Finite Automata and Petri nets - Simulation of Processes in Logistics", in *ECMS 2024: 38th International ECMS Conference on Modelling and Simulation*, D. Grzonka, N. Rylko, and G. S. V. Mityushev, Eds., Krakow (Poland), 2024, pp. 474–480.

[4] C. Simon, S. Haag, and L. Zakfeld, "The Process-Simulation.Center", in *SIM-SC: Special Track at SIMUL 2022: The Fourteenth International Conference on Advances in System Simulation*, F. Herrmann, Ed., Lisbon (Portugal), 2022, pp. 74–77.

[5] T. Bergs *et al.*, "The Concept of Digital Twin and Digital Shadow in Manufacturing", *Procedia CIRP*, vol. 101, pp. 81–84, 2021, 9th CIRP Conference on High Performance Cutting, ISSN: 2212-8271.

[6] S. Russell and P. Norvig, *Artificial Intelligence - A Mordern Approach*, 4th ed. London: Pearson, 2021.

[7] P. Flach, *Machine Learning - The Art and Science of Algorithms that Make Sense of Data*, 9th ed. Cambridge: Cambridge University Press, 2012.

[8] J. Howard and S. Gugger, *Deep Learning for Coders with fastai & PyTorch*. Sebastopol: O'Reilly, 2020.

[9] A. Géron, *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow*. Sebastopol: O'Reilly UK Ltd., 2019.

[10] C. Mattmann, Ed., *Machine Learning with TensorFlow*, 2nd ed. Shelter Island, NY: Manning, 2020.

[11] M. Karatas, *Development of AI applications (in German: Eigene KI-Anwendungen programmieren)*. Bonn: Rheinwerk Computing, 2024.