Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

SIMULATION-BASED DIGITAL TWIN OF A COMPLEX SHOP-FLOOR LOGISTICS SYSTEM

Dávid Gyulai
Júlia Bergmann

Centre of Excellence in Production Informatics and Control (EPIC)
Institute for Computer Science and Control (SZTAKI)
Kende 13-17
Budapest 1111, HUNGARY

Attila Lengyel

Digital Analytics Office
Western Digital Corporation
5601 Great Oaks Pkwy
San Jose, CA 95119, USA

Botond Kádár
Dávid Czirkó

EPIC InnoLabs Ltd.
Kende 13-17
Budapest 1111, HUNGARY

ABSTRACT

Digital analytics tools have been at the forefront of innovation in manufacturing industry in recent years. To keep pace with the demands of industrial digitization, companies seek opportunities to streamline processes and enhance overall efficacy, opting to replace conventional engineering tools with data-driven models. In a high-tech factory, detailed data is collected about the products, processes, and assets in near-real time, providing a basis to build trustworthy analytical models. In this paper, a novel discrete-event simulation (DES) model is proposed for the detailed representation of a complex shop-floor logistics system, employing automated robotic vehicles (AGV). The simulation model is applied to test new AGV management policies, involving both vehicle capacity planning and dispatching decisions. In order to illustrate the usefulness of the model and the effectiveness of the selected policy, numerical results of a case-study are presented, in which the selected policy was realized in a real manufacturing environment.

1 INTRODUCTION AND MOTIVATION: MANAGEMENT OF AGV FLEETS IN PRODUCTION

In the age of digital manufacturing, cyber-physical systems, lights-off factories, and process automation prove crucial, as the latest technologies - including both software and hardware solutions - offer tremendous potential to increase effectiveness in manufacturing. In recent decades, automation efforts have not only concentrated on manufacturing processes, but also on production logistics. Applying conveyor systems and autonomous mobile robots in logistics, the raw materials and finished goods can be transported within production facilities with ease. The design and management of these systems, however, requires careful engineering decisions due to the high associated costs and performance-critical impact. Focusing on free-range transportation in a fully automated and flexible production system, the continuous operation of the machines can be maintained only by accurate vehicle fleet management, otherwise the overall system output and utilization will drop. Modern automated guided vehicles (AGVs) are driverless, free-range transport devices that carry multiple items on the shop floor, apply automatic control, and typically operate in a

fleet (Franke and Lütteke 2012). Typical decisions associated with the AGV fleet management include fleet capacity planning, task determination, delivery dispatching, pickup dispatching, load selection, and route planning (Ho, Liu, and Yih 2012). In this paper, fleet capacity planning and dispatching decisions are investigated with the goal of maximizing the machines' utilization and production performance.

While both of the aforementioned decisions can be supported by constrained optimization and simulation tools, the applied methods should be carefully selected given the running time of the algorithms and the quality of the final solution (Ham 2019). In a fully automated plant with a large number of machines, the applied vehicle fleet may consist of several vehicles. Although vehicle scheduling and constrained vehicle routing methods exist to solve the associated problems optimally, their application may not be possible in a highly dynamic production environment with uncertain conditions and short expected calculation times. These methods typically require the decision makers to foresee the operations and tasks, which can prove challenging, or even impossible, in a complex manufacturing environment, e.g. cyclic schedule of an AGV fleet can be rarely applied in such cases (Bocewicz, Nielsen, and Banaszak 2014).

Leveraging simulation technologies might be the right track to support AGV fleet management decisions, as they can be applied to find the proper fleet sizes and most promising control logics, even in a very complex production environment. Typically, the related approaches are applied in an offline way: a series of experiments is performed to find the promising settings, and then the identified solutions need to be realized to bring business value. Simulation-based approaches exist to solve AGV path planning (Muller and Cardinal 2002), vehicle guiding (Duinkerken, ter Hoeven, and Lodewijks 2006), vehicle routing (Klaas, Laroque, Dangelmaier, and Fischer 2011; Faulin, Gilibert, Juan, Vilajosana, and Ruiz 2008), and scheduling (Wang and Zhou 2015) problems. However, only a few approaches exist for fleet capacity planning and vehicle dispatching (Ho and Chien 2006; Li and Kuhl 2017), especially assuming a highly complex and dynamic manufacturing environment with a large quantity of machines and uncertain parameters.

The purpose of the presented research is to employ a highly detailed discrete-event simulation model to identify the most promising task determination and dispatching policies for a fleet of AGVs operated in a complex manufacturing environment. A hierarchical approach was applied to decompose the overall problem into task assignment (i.e. planning) and dispatching decisions (i.e. execution). In addition to the simple task assignment mechanisms, an innovative network analytics solution was also proposed to define the operating zones of the vehicles. Capturing the underlying links among the elements of the material flow network, a high-performance yet computation-hungry logic was identified, therefore facilitating deployment of a simplified algorithm in the real manufacturing environment.

2 PROBLEM STATEMENT

This section specifies the analyzed problem in detail. As for the physical environment, a large shop-floor environment is given, operating several *machines*. The machines are responsible for completing the *jobs* under the *cycle time*. A machine can start to operate only if all of its input ports are loaded with the physical *items* required for the job. After the job is finished, the completed items appear on the output port of the machine. Besides the active job processing units, there exist a passive buffer and storage objects in the material flow; these, however, are only responsible for holding the items that they receive. Among those objects of the material flow, AGVs implement the transportation of the items, i.e. they complete the transportation *tasks* that belong to machine (un)loading. By nature, every item has two corresponding tasks: a *pickup* and a *delivery* task, that inherently need to be completed by the same vehicle. Each vehicle thus maintain its own list of tasks, comprised solely of the delivery tasks that have already been picked up. Accordingly, each task is specified by (1) the item to carry; (2) its type, which is either delivery or pickup; and (3) its destination or source station respectively.

The AGVs are controlled by a central dispatcher that is responsible for calculating the task-vehicle assignment. At any given point in time, an AGV is executing a single task only and requests a new task immediately after completion of the original task. Every production has its predefined routing, which determines the transportation tasks to be completed. Assuming a continuous run of production, machines

trigger the transportation task requests periodically towards the vehicles. The time between the task triggers is called machine cycle time, and is predefined in the job routing. Every job completion is finished within the cycle time and, thereafter, tasks for new item loading and previous item removal are triggered. Machines can only start the job processing if all of the loading tasks are completed. In the event that the loading actions are not completed and a machine has already finished processing the previous job, the machine goes to a *waiting* state until the loading completes. Conversely, when a machine has finished processing the job but the previous unloading task has not yet completed, the machine will enter into a *blocked* state until an AGV performs the unloading. Accordingly, the objective is to ensure the continuous run of machines via proper (un)loading services performed by the AGVs, and any production loss due to imperfect logistics is accounted for by the AGV system. Due to the central management of tasks and vehicles, the task manager maintains full access to the list of tasks and can also share them with any vehicle.

All vehicles are identical, and they have a capacity for transporting multiple parts. The task handling is completely automated by a robotic arm installed on every vehicle. Between the stations, a route network exists that imposes physical constraints on the vehicle movements. Among these constraints, the most typical ones are the route directions and the width of the corridors. Altogether the results require that the AGVs need to approach the stations with the right orientation (i.e. forward or backward) in order to reach the machine ports with the robotic arm. Depending on the execution sequence of tasks, the approach orientation towards a given station might be different; thus proper task sequencing helps to reduce the vehicle's movement times.

Given the above constraints and the dynamic nature of the incoming stream of tasks, the AGV fleet as a whole is responsible for providing a transportation service which accomplishes each task in a way that minimizes total losses on a given horizon. In a saturated system—typical in semiconductor manufacturing—this implies that the objective of maximizing the utilization U of active machine stations. Additional KPIs include the total number of tasks n completed in a given period of time, as well as the task duration t_d that spans between the task triggering and finishing time points. The AGV motion controller supports the physical operation of vehicles, like localization or collision avoidance, whereas the workflow suggested below greatly alleviates the prevention of collisions and deadlocks.

3 SIMULATION MODEL ARCHITECTURE AND FEATURES

The first step in optimizing the AGV system is to understand the system dynamics and identify the opportunities and weak points that enable improvements in the AGV control logic. A detailed simulation model, therefore, needs to be built, representing all of the aforementioned features and constraints. The architecture of the overall system simulation model and the details of the applied object models are provided in the sections to follow.

3.1 Description of the model architecture

Similar to the real system, the simulation model is also comprised of two main parts: the model of the physical system, including the machines and AGVs *(simulator)*, and the model of the AGV controller and task dispatcher *(dispatcher)* (Figure 3.1). This architecture is applied for two main reasons: on one hand, it provides higher modeling flexibility, as any of the two parts can be replaced with other representations, even with the real systems; on the other hand, it enables the concurrent development of the model that significantly boosts the modeling cycle. As the communication is established via TCP/IP socket interface, the controller and the emulator do not necessarily run on the same computer.

The system model includes all of the physical objects of the shop-floor, with detailed representations of the route network, the AGVs, and the machine operation logic. This part of the model works in a slave mode, which means that both AGV transportation and task execution movements as well as the job processing actions are simulated in this model; new actions and their sequences, however, are determined by the controller. After a task is executed (e.g. a machine or an AGV completes a task), the simulation

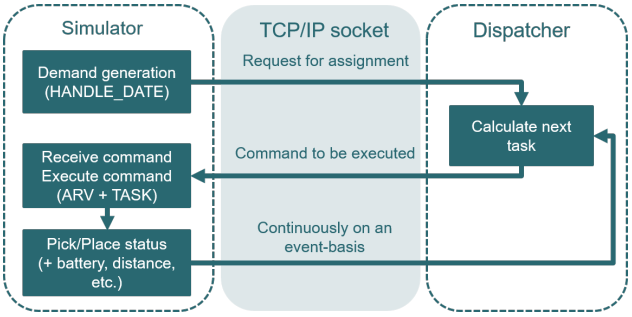

Figure 1: Architecture of the system's DES model, with the simulator and dispatcher sub-models and the TCP/IP socket communication between them.

clock is paused and a notification message is sent to the controller. The simulation will be halted until the controller sends back the next command to be executed.

The system controller, by contrast, is responsible for processing the notification messages received and calculating the next actions to be executed, with the aim of improving the KPIs. Accordingly, the controller should always have an overview of the actual status of the machines and vehicles. As for the latter, the cargo sizes, battery levels and charging requests, and the last known positions need to be maintained in a data set. Furthermore, the machines' coordinates, parking positions' status, and the work-in-progress also need to be followed and handled in dispatching decisions. Messages (e.g. request of status update) are generated by the vehicles and machines, and calculated actions are sent to the AGVs (one at a time). If there is no new action to be performed (e.g. there is no free vehicle to assign to a task), the task is stored in the central task list.

3.2 Description of the object models

In this section, the main elements of the AGV simulation are detailed from a technical perspective. Even though several features of the applied DES software tool were utilized to build up a realistic model of the system, many custom objects and logics are implemented to model the environment in detail. As for the modeling phases, the route network was built up first, applying the exact coordinates and scale of the real system, furthermore, representing the physical barriers and route directions (one- or bidirectional). Accordingly, there were some routes which could only be approached by the vehicles in a reverse direction in order to serve the machines. If a vehicle had to switch directions, the related event was handled before entering the given lane, and the vehicle was rotated under a given time. The second biggest challenge was to address deadlock situations in the system resulting from vehicles blocking each other's movements, as this would result in cessation of the entire simulation after a brief period of time. Several random vehicle movements were therefore executed with a large number of vehicles to identify the critical junctions in the system, and the traffic rules in these points were handled individually.

Once the route network modeling is completed, the machines' simplified models are implemented. The machines' logic describes the loading, unloading, and operational states as well as the related transition in any possible state that can happen during operation. The machines' processing, loading, and unloading times follow predefined distribution functions. The last main objects in the simulation model are the vehicles and the application of real velocity, capacity, and dimensional attributes. Since AGVs have imperfect characteristics, random alarms are simulated by using the availability and mean time to repair parameters. An important feature of autonomous mobile robots is the battery life, hence the energy consumption and discharging characteristics have been modeled by using the *A/h* parameters for driving, loading, and parking states. Any time the charge level reached a certain threshold, the vehicle was given a top priority task to visit the battery station and replace the discharged battery with a full one. If a vehicle do not have any task

assigned, it is sent to the nearest parking spot to wait until the next command. In this way, the continuous operation of the overall fleet is guaranteed in the simulation.

4 ANALYTICS AND OPTIMIZATION TOOLSET FOR AGV CONTROL

The simulation model of the system is applied to assess different control logics and production scenarios in order to identify the best logic and its parameter settings for deployment. The logics are implemented in the controller model, and the system model is used as a testbed for evaluation.

4.1 Vehicle fleet planning and high level task assignment

Supporting the vehicle assignment, various filtering rules are used to prepare the dispatching decisions. Applying a hierarchical problem decomposition, the overall decision workflow is split into two main parts to boost the calculations and simplify the dispatching decisions. On the higher level, called the *planning level*, the solution space for vehicle tasks-assignment is narrowed by high-level rules, taking into account the task and AGV attributes, machine, and task locations. Based on these factors, three main rules are defined with different levels of effectiveness and implementation complexity.

The first rule—called *CollectOrDistribute*—relies on the separation of pickup and delivery tasks. Accordingly, some vehicles are dedicated towards machine loading, and some are dedicated towards unloading. This separation guarantees that the overall task list can be halved, thus the real-time dispatching decisions may be significantly boosted. Even though this presumes a good performance and ease of implementation, it may only work well in case of relatively simple routings with identified source and destination locations (i.e. buffers), and only a few—ideally, a single—processing units in between.

The second rule—called *JobType*—assigns the tasks to vehicles based upon the job types. AGVs are dedicated to perform a single task type only. This means that, independently from the other attributes such as task locations and pickup or placement, an AGV can receive a task if the main task attribute (e.g. item type, routing class) criteria is met. If there are more than two classes, the task list can be significantly reduced by type-based filtering, thus enabling efficient pre-processing to boost the the dispatching decisions. However, this requires the stability of the production in terms of product mix, and/or a high flexibility of the control system is needed to reassign AGVs to new job types to maintain vehicle utilization in case the product mix would change.

As a third and most advanced approach, a novel clustering-based approach is proposed—called *Zones*—that assigns the stations (i.e. machines and buffers) to non-overlapping shop-floor zones. Similarly, the vehicles are also assigned to specific zones, and they can execute a task only when the pickup/placement locations match the predefined AGV zones. Although the zone definition can be solved with various clustering approaches (e.g. k-means or hierarchical), here a solution is proposed that derives benefits not only from the static, e.g. spatial features of the system, but is also able to capture the interaction (i.e. material flow links) among the machines and buffers.

As the fist step of the proposed approach, a network analytics problem is solved. The material flow network is represented as a graph, $G = G(V,E)$, where nodes $V = \{v_i\}_{i=1}^{N}$ are the machines and buffers, while edges $E = \{(v_i, v_j) | A_{ij} = 1\}$ are the directed edges representing the possible movements of items performed by the vehicles. The weight of the directed edges is calculated as the combination of the intensity of material flow between the objects and their distances, either in space or time. In order to optimize the assignment of AGVs to clusters of workstations, an optimal clustering of the nodes must first be found. Node clustering is defined as an optimization problem, which results in the best segmentation of the shop-floor by using the transportation intensity graph. In an ideal setting, an AGV is completing tasks in a manner that minimizes travel time in order to maximize the number of tasks that it can complete. Minimal travel distances between pickup and delivery points can be achieved by assigning the tasks to an AGV based on spatial attributes. In order to achieve this, node clusters are defined by maximizing the strength of division of the network into modules. The *modularity* of a network with given division is a

measure of the structure of networks. Networks with high modularity have dense connections between the nodes within modules, but sparse connections between nodes in different modules. Node clustering is achieved by solving the graph modularity maximization problem to follow.

Let $A = \{A_{ij}\}_{i,j=1}^{N,N}$ represent the adjacency matrix of G with $a_{ij} = 1$ if $(v_i, v_j) \in E$, otherwise 0. The out-degree and in-degree of node v_i are defined as $k_i^- = \sum_j A_{ij}$ and $k_i^+ = \sum_j A_{ji}$ respectively. Since the modularity calculation was originally defined for undirected graphs (Newman 2006), a slight modification is proposed to extend it for directed graphs. Let D_{ij} be the shortest distance between v_i and v_j, and I_{ij} be the material flow intensity between v_i and v_j, which is predicted from historical data. The material flow intensity is determined by the average number of tasks to be transported between the nodes per unit of time. In the absence of historical data, this value can be predicted by using the machines' average cycle times.

Given this, the weight for an edge between v_i and v_j is defined as $B_{ij} \doteq A_{ij} I_{ij} D_{ij}^{-1}$, which forms a new adjacency matrix of an edge-weighted directed graph, expressing the transportation intensity between two nodes. This intensity is proportional to the distance between the nodes and inversely proportional with the material flow intensity between them. Also, the overall edge weight and the out- and in-degrees of the nodes must be updated with respect to the new adjacency matrix, i.e. $m \doteq \sum_{i,j} B_{ij}$, $k_i^- \doteq \sum_j B_{ij}$ and $k_i^+ \doteq \sum_j B_{ji}$. The modularity (i.e., quantity of the strength of a community structure) of a clustering $\mathscr{C}$ of a directed graph G is defined as

$$Q(\mathscr{C}) = \frac{1}{2m} \sum_{ij} \left(B_{ij} - \frac{k_i^- k_j^+}{2m} \right) \delta(c_i, c_j) \tag{1}$$

The modularity of a clustering $\mathscr{C}$ quantifies its strength. Accordingly, $Q(\mathscr{C}_1) > Q(\mathscr{C}_2)$ means that $\mathscr{C}_1$ is a better clustering on G graph subject to gathering vertices into groups such that there is a higher density of edges within groups than between them, accordingly, a trivial clustering is when every node has its own cluster. Finding the strongest clustering on a graph is equivalent to finding the $\mathscr{C}^*$, which maximizes $Q(\mathscr{C})$ modularity function. Formally, modularity maximization can be captured by an integer programming model (IP), as follows:

$$\text{maximize} \quad \frac{1}{2m} \sum_{ij} \left(B_{ij} - \frac{k_i^- k_j^+}{2m} \right) X_{ij} \tag{2}$$

$$X_{ii} = 1 \qquad \forall i \in V \tag{3}$$

$$X_{ij} = X_{ji} \qquad \forall (i,j) \in V^2 \tag{4}$$

$$\begin{cases} X_{ij} + X_{jk} - 2X_{ik} \\ X_{ik} + X_{ij} - 2X_{jk} \\ X_{jk} + X_{ik} - 2X_{ij} \end{cases} \leq 1 \qquad \forall (i,j,k) \in V^3 \tag{5}$$

In the above problem, objective function (2) maximizes the graph modularity, applying the binary variable X_{ij}, which equals 1 if nodes i and j belong to the same cluster, and 0 otherwise. Modularity maximization is computationally intractable (Dinh, Li, and Thai 2015); given this, in case of transportation networks with a large number of edges and dense connections, finding the exact optimal solution via classical optimization approaches may not be possible. Hence, heuristics are applied to find good clustering; among those heuristic methods employed, greedy, spectral, and global search methods are the most commonly applied techniques (Chen, Kuzmin, and Szymanski 2014). For the sake of simplicity, a greedy algorithm is applied for the vehicle fleet planning, as described by the Algorithm 1.

Once the zones have been defined, another related problem to be solved is the assignment of vehicles to zones. This leads to a load balancing, which can be completed by distributing the vehicles among zones proportionally to the number of machines in the zones. It is important to underscore that this may lead to unbalanced loads from a fleet perspective; thus the assignment can be iteratively refined by increasing the operating zones per vehicle. Alternatively, the load balancing can be formulated as a quadratic

Algorithm 1 Greedy algorithm of modularity maximization

1: Given $G = G(V,E)$ directed graph with weighted edges by B adjacency matrix as in (1)
2: Given $\mathscr{C}$ initial trivial clustering with $n = |V|$ number of clusters (partition into singletons)
3: **while** $n > 1$ **do**
4: **for all** cluster pairs in $\mathscr{C}$, calculate the modularity $Q(\mathscr{C}_{ij})$, where $\mathscr{C}_{ij}$ is derived from $\mathscr{C}$ by merging i and j clusters **do**
5: $\mathscr{C} \doteq \underset{c_z}{\operatorname{argmax}}\,(Q(\mathscr{C}_z))$
6: $z = z - 1$
7: **end for**
8: **end while**
9: **return** $\mathscr{C}$ clustering with the highest modularity

integer optimization problem, though its optimal solution can only be obtained by expending significant computational effort.

4.2 Vehicle dispatching and task execution

The previously described fleet planning methods are applied as part of the preprocessing of the simulation experiments, which means that the high level vehicle role description can be performed without using the simulation tool. However, the results of these preprocessing steps provide important input for the simulation, as the solution space of the possible vehicle-task assignments can be significantly narrowed down by using the zone or role restrictions. In the simulation model, the actual task sequencing and execution control is performed by using dispatching rules. These sets of rules can be applied to determine the next action for a given AGV based upon vehicle status, as well as spatial and time considerations.

The dispatcher function is the core part of the simulation model and is called in two different cases. In case of the workstation initiated rule (i), when a new task is triggered by a machine, a vehicle is sought to execute it. In the opposite case called as workstation initiated rule (ii), a new task is to be selected for a vehicle that became free after the execution of the previous task. In saturated production systems where machine capacity is the bottleneck and jobs are waiting to be processed, the latter case occurs more often, i.e., tasks are perpetually pending in the task list and anytime when a vehicle has completed a task, it can select from these tasks, while it is rare that any vehicle is free/idle at a triggering time of a new task.

In the simulation experiments, two different dispatching rules are applied: the first-in-first-out (FIFO) rule and a combined distance and time based (DTB) ruleset. Given to the aforementioned reasons, the vehicle-initiated part of the algorithm is detailed by Algorithm 2. The FIFO rules assign the first possible item of the task list to a vehicle, considering the assignment conditions provided by the planning decisions, e.g. zones. Here, it is assumed that the task list is ordered by the trigger time of the tasks, i.e., tasks that were triggered earlier are in the top of the task list, while the new ones are pasted in the end of the list. In contrast, the DTB rules try to find the right balance between the trigger time, the vehicle's actual position, and the future positions based upon the tasks that are already picked up. According to the terminology presented by Ho and Chien (2006), the algorithm implements a pickup-task-first dispatching, which means that vehicle loading is prioritized over machine loading. In case of multiple-load AGVs, this leads to higher vehicle utilization and less commuting between the machines and buffers.

All parameters in Algorithm 2 are monitored and known by the AGV control system at any time: $D(x,y)$ denotes the distance between locations x and y, P_i is the location of i, X_i is the zone of i, and R_i is the criticality factor of task i, characterizing the remaining time until the task completion due time. For task assignment, *FilteringConditions* are applied and, depending on the task assignment, the logic may rely on zone assignment and task types of collect/distribute conditions.

Algorithm 2 Vehicle-initiated DTB algorithm

if *TaskPrio* == "PickUp" **then**
  for all task q in the task list **do**
    if $TaskType(q)$ == "PickUp" **and** $FilteringConditions$ == $TRUE$ **then**
      for all task r in $AssignedPlacements$ **do**
        $f_q = \min\left(D\left(P_q, P_r\right); D\left(P_r, P_q\right)\right)$
        $e_q = w_d \cdot D\left(P_{AGV}, P_q\right) + w_{lt} \cdot R_q$
        $c_q = e_q + f_q$
      end for
    end if
    return $q = \underset{q}{\operatorname{argmin}}\ c_q$
  end for
else if *TaskPrio* == "Placement" **or** q == void **then**
  return q = $q = \underset{q \in AssignedPlacements}{\operatorname{argmin}} D\left(P_{AGV}, P_q\right)$
end if

5 EXPERIMENTAL RESULTS

The DES model and the related analytics algorithms are implemented within a real industrial case study with the objective of increasing the operational performance of the system under study.

5.1 Use case description

A real system from the semiconductor industry was used as a test environment to assess the AGV control logics. Due to the high production load, only the simulation model of the system was applied to analyze the logics and benchmark them with the aim of realizing the most promising solution. The real production system and the corresponding processes follow a job-shop scheme, including almost 200 machines and several buffers. In addition to the temporary storage places on the shop floor, every machine had input and output buffers to make continuous operation easier. In the analyzed scenarios, the vehicle fleet consisted of 14 to 18 vehicles, all having the same attributes in terms of such attributes as speed or capacity. Every AGV had uniform control properties as well, and the controller had full visibility of the fleet, enabling a centralized dispatching scheme. The system is typically saturated with jobs, which means there are always tasks (work-in-progress) waiting for machines to become free to load. The AGV route network is relatively complex, consisting of several one- and bi-directional routes, and also constraints that may require vehicles to approach the machines with either forward or backward movements due to the system and AGV vehicles' designs. From a simulation perspective, modeling these physical constraints and avoiding *deadlock* states were real challenges that could be tackled only by detailed model building and custom-defined model objects and attributes. Important to underscore is that, during the launch phase of the analytics project, the overall logistics system was semi-automated, which means that in the event of an AGV capacity shortage, operators performed the task execution, specifically the machine loading and unloading, to avoid idle machine times. However, for the sake of comparability and to avoid distortions in the evaluation, manual task completions were not included in the DES model, thus machine downtime was often observed.

5.2 Assessment of the AGV logics by using simulation

As the real environment could not be used to analyze and implement all of the candidate logics without significant production losses and downtime, the implemented simulation model played a key role in the overall decision support toolset. However, the preprocessing items—especially the models that are responsible for the vehicle fleet planning—are implemented in separate applications, as their analytics time

frame and calculation steps differ from those of the simulation. Although the *CollectOrDistribute* and *JobType* filtering rules can be easily implemented in any DES tool, the *Zones* relies on a greedy algorithm that can be implemented in C++ in a more efficient way. All other calculations are performed directly in the DES model that was implemented in *Siemens Tecnomatix Plant Simulation*.

The simulation model was validated by using historical AGV logs, which in total included 14,000 AGV tasks. The validation KPI was the average task duration, which quantifies the accuracy of the AGV vehicle parameters and the vehicle routing. The overall duration of a single task is measured from the task start time until the task completion; computation in this manner thus includes the travel time and the item handling time at a given machine. Additionally, the number of daily task completions was also considered, and with both measurements, the average relative values were applied to determine the model accuracy. This validation procedure, however, does not qualify or measure the sequence of task executions; instead, this was tackled by forcing the simulation to follow the real historical sequence, thus the dispatching logic was switched off in this case. According to the numerical validation tests, the model was 94% accurate after refinements, which is satisfactory considering the high level of complexity and dynamics of the system.

Accepting the simulation model to be valid and fixing the AGV and machine parameters, only the control logics and fleet sizes were changed in the subsequent phases of the analytics work. Each task assignment logic has been encoded in the model and evaluated within a comprehensive benchmark. Supported by the few possible combinations of the input variables, a full factorial experiment was designed and executed first. In this case, the vehicle fleet size was changed between 14 to 18, which lead to 15 different scenarios (3 logics, 5 fleet sizes); furthermore, 10 experiments were performed in all cases with different random seed settings. The simulation time span was 3 days, and a single experiment took around 3 minutes on average. The results of the first experiments are summarized in Table 1. Per the results, the KPI values are relative to each other and scaled to ensure that the best value always corresponds to 100%.

Table 1: Benchmark results of the vehicle fleet planning methods. Each value is the average of 10 experiments with different random seed values; results are scaled and 100% belongs to the best setting in each column. Average values of total completed tasks, machine utilization, and task duration are denoted by $\overline{n}$, $\overline{U}$ and $\overline{t_d}$, respectively

	$\overline{n}$ [%]	$\overline{U}$ [%]	$\overline{t_d}$ [%]
CollectAndDistribute	91%	80%	93%
TaskType	87%	79%	92%
Zones	100%	100%	100%

According to the results, the best performance could be achieved by using the *Zones*; other simple approaches also performed relatively well, however. In the second round of experiments, the emphasis was put in the assessment of dispatching logics, comparing the FIFO and DTB rules specifically. Furthermore, the impact of modularity-based zone definition was also assessed by comparing it with a hierarchical clustering approach. A significant difference between these zoning methods is that the latter may not capture the strength of transportation links among the machines; furthermore, the number of clusters are also not provided directly and, therefore, needed to be arbitrarily selected after the cluster calculation. In these experiments, only the *Zones* logic was used for fleet planning, and the application of different random seed settings, two clustering and dispatching approaches, resulted in 40 total experiments. The results of the first experiments are summarized in Table 2.

On the one hand, modularity-based clustering outperforms the hierarchical clustering in task completions and also in average machine utilization. On the other hand, DTB dispatching provides much better results than the simple FIFO rule. Conclusively, the combination of these two approaches leads to the best overall system performance in the analyzed cases. However, it is important to highlight that in the case of complex systems, both methods require significant computational effort. For example, in the case of the modularity

Table 2: Benchmark results of the dispatching rules and zone definition approaches (hierarchical and modularity based clustering) methods. Each value is the average of 10 experiments with different random seed values; results are scaled and 100% belongs to the best setting in each column.

Zoning	Dispatching	$\overline{n}$ [%]	$\overline{U}$ [%]	$\overline{t_d}$ [%]
Hierarchical	DTB	73%	80%	94%
Hierarchical	FIFO	63%	61%	74%
Modularity	DTB	100%	100%	100%
Modularity	FIFO	77%	77%	69%

calculation, the greedy algorithm can take hours to run[1]; the distance-based dispatching approaches, by contrast, seek the minimal distance in every task assignment calculation, which requires looping over the list of open tasks twice.

5.3 Realization of the selected logic

After assessing all of the possible scenarios evaluated by using simulations, the results were shared with the operations team to identify the logic that brings the most business value. The logic is expected to have high associated system performance and be easily deployable in a shop-floor dispatching system. Due to its simple fleet planning mechanism, and good, stable resulted performance, *CollectAndDistribute* logic was selected as the new high-level task assignment rule. As for the dispatching rules, although the DTB approach highly outperformed the FIFO logic, only the simplified version of the former was implemented. According to the computational tests with the real fleet controller, full implementation of the ruleset would require to high computational efforts and generate unstable operation. The simplified version selected for implementation applies some static parameters to find open tasks near the vehicle. Even though the traveled distance might not always be the shortest, the running performance of the system is more stable due to the robust implementation.

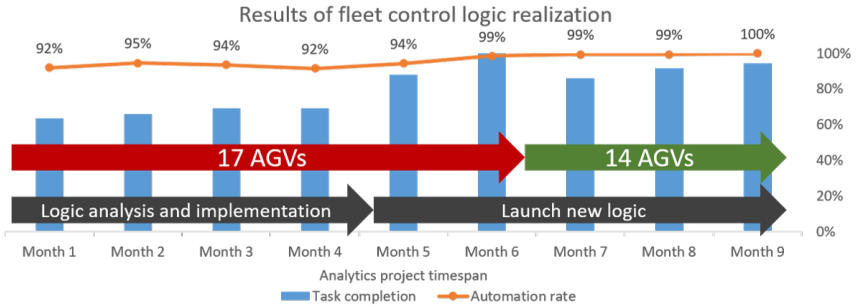

Figure 2: AGV system performance over the the time-span of the analytics project, with all results provided in percentages relative to the best value. Aside from the increase in the automation rate and the number of task completions, the new logic enables to maintain stable system operation with three vehicles less than previously.

After encoding the new algorithms in the fleet controller software, the real shop-floor automation have been switched to utilize the new methods, and the KPI values were observed continuously, monitoring the system stability during the roll-out phase. After recognizing the best parameter settings, the overall shop-floor was switched to use the new algorithms. Since the launch of the new system, the AGV fleet is operating as expected and with a higher overall performance, as summarized in Figure 5.3.

[1]The running time was 10 minutes on average in the analyzed cases.

6 CONCLUSION AND OUTLOOK

According to the results, the simulation-based analytics project significantly contributed to an increase in the performance of a complex production system that is heavily loaded with jobs. The greatest overall benefit of applying simulation in this case was the opportunity to assess various new ideas without disturbing the real system. Although the model building and system analysis took additional months to complete, the results demonstrate a rapid return on the investments associated with the analytics project. Important to highlight is that the most innovative, yet highly complex *Zones* rule with the modularity-based clustering approach was proven to outperfrom all the other logics. In reality, however, the implementation of the logic would require much higher computational performance compared to the second best approach. Although even complex logics may be implemented with relative ease in a simulation model, real production environments with communication and control overload require efficient and simple approaches to guarantee their stability and high performance.

As a result of the successful application of a digital twin approach in the presented use case, similar analytics projects have been launched to increase various systems' performance. Furthermore, as a part of the future work, the authors plan to conduct a more detailed investigation of modularity-based clustering applications in production environments, as the advantage of network analytics in manufacturing environments is visible based upon the analytics results contained herein. In addition to the proposed distance-based dispatching approach, new solutions relying on online scheduling and optimization are sought by using constrained programming solutions and heuristics-based approaches.

REFERENCES

Bocewicz, G., I. Nielsen, and Z. Banaszak. 2014. "Automated guided vehicles fleet match-up scheduling with production flow constraints". *Engineering Applications of Artificial Intelligence* 30:49–62.

Chen, M., K. Kuzmin, and B. K. Szymanski. 2014. "Community detection via maximization of modularity and its variants". In *IEEE Transactions on Computational Social Systems*, Volume 1, 46–65. IEEE.

Dinh, T. N., X. Li, and M. T. Thai. 2015. "Network clustering via maximizing modularity: Approximation algorithms and theoretical limits". In *2015 IEEE International Conference on Data Mining*, 101–110. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Duinkerken, M. B., T. ter Hoeven, and G. Lodewijks. 2006. "Simulating the operational control of free ranging AGVs". In *Proceedings of the 2006 Winter Simulation Conference*, edited by L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 1515–1522. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Faulin, J., M. Gilibert, A. A. Juan, X. Vilajosana, and R. Ruiz. 2008. "SR-1: A simulation-based algorithm for the capacitated vehicle routing problem". In *Proceedings of the 2008 Winter Simulation Conference*, edited by S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, and J. W. Fowler, 2708–2716. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Franke, J., and F. Lütteke. 2012. "Versatile autonomous transportation vehicle for highly flexible use in industrial applications". *CIRP Annals* 61(1):407–410.

Ham, A. 2019. "Transfer Robot Task Scheduling in Semiconductor Manufacturing". In *Proceedings of the 2019 Winter Simulation Conference*, edited by N. Mustafee, K.-H. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, 2248–2256. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Ho, Y.-C., and S.-H. Chien. 2006. "A simulation study on the performance of task-determination rules and delivery-dispatching rules for multiple-load AGVs". *International journal of production research* 44(20):4193–4222.

Ho, Y.-C., H.-C. Liu, and Y. Yih. 2012. "A multiple-attribute method for concurrently solving the pickup-dispatching problem and the load-selection problem of multiple-load AGVs". *Journal of manufacturing systems* 31(3):288–300.

Klaas, A., C. Laroque, W. Dangelmaier, and M. Fischer. 2011. "Simulation aided, knowledge based routing for AGVs in a distribution warehouse". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 1668–1679. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Li, M. P., and M. E. Kuhl. 2017. "Design and simulation analysis of PDER: A multiple-load automated guided vehicle dispatching algorithm". In *Proceedings of the 2017 Winter Simulation Conference*, edited by W. K. V. Chan, A. D'Amborgio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, 3311–3322. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Muller, D. J., and S. M. Cardinal. 2002. "Complexities of AGV modeling in newspaper roll delivery system". In *Proceedings of the 2002 Winter Simulation Conference*, edited by E. Yiicesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, 1046–1051. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Newman, M. E. J. 2006. "Modularity and community structure in networks". In *Proceedings of the national academy of sciences*, Volume 10323, 8577–8582. National Acad Sciences.

Wang, M. C., and Y. Zhou. 2015. "Scheduling for an automated guided vehicle in flexible machine systems". In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, 2908–2916. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

DÁVID GYULAI is a research fellow at the Institute for Computer Science and Control (SZTAKI). He obtained his MSc (Mechatronics Engineering) and PhD degrees at Budapest University of Technology and Economics (Hungary). He has previously been involved in several research and development projects with Hungarian and multinational companies. He is interested in digital enterprise technologies, production and capacity planning, and data analytics. Since 2018, he has been a research affiliate of the International Academy for Production Engineering (CIRP). His email address is david.gyulai@sztaki.hu.

JÚLIA BERGMANN is a data analyst at EPIC InnoLabs Ltd. and a research associate at SZTAKI. She obtained her MSc degree (Mathematics) at Budapest University of Technology and Economics. In the past, she has been involved in industrial projects on the support of management control, adaptive network analysis for planning and scheduling, and she has also worked in several research projects. She is interested in visual analytics, statistics, and machine learning. Her email address is julia.bergmann@sztaki.hu.

ATTILA LENGYEL is the director of Strategy, Innovation & Adoption at the Digital Analytics Office of Western Digital Corporation (WDC). At WDC, he led the development and implementation of key business solutions at global factory sites. He previously worked and lived in Japan, leading an Operations Research team at Hitachi Ltd. in Yokohama. Before joining Hitachi Ltd., he studied at Tokyo University and Kobe University (Japan), earning his PhD in Systems Science in 2004.

BOTOND KÁDÁR is the managing director of EPIC InnoLabs Ltd. and former senior researcher at SZTAKI. He obtained his M.Eng. and PhD degrees at Budapest University of Technology and Economics. Since 2010, he has served as an associate member of CIRP, and is currently a member of the International Federation of Automatic Control (IFAC). He is involved in several research and development projects in fields of interest to him: production control, simulation, and multi-agent approaches for production engineering. His email address is botond.kadar.@epicinnolabs.hu.

DÁVID CZIRKÓ is a systems engineer at EPIC InnoLabs Ltd. He obtained his MSc (Mechatronics Engineering) at Budapest University of Technology and Economics in 2016. He has previously been involved in several industrial projects spanning various domains, including industrial automation, semiconductor manufacturing, and internal logistics. He is interested in digital enterprise technologies, production analy-tics and optimization, as well as material flow simulation. His email address is david.czirko@epicinnolabs.hu.