Mathematical model for factory layout and production systems balancing

Deliverable 5.3

MUWO

MULTI-METHOD WORKSPACE FOR HIGHLY SCALABLE PRODUCTION LINES

Project identifier	MUWO
Project title	Multi-method workspace for highly scalable production lines
Document version	V1.0
Planned delivery date	M24 (January 2023)
Actual delivery date	M24 (31/01/2023)
Document title	Mathematical model for factory layout and production systems balancing
Work Package	WP 5
Abstract	This deliverable provides a mathematical description of the models and algorithms used in the project for resource optimization, production scheduling, MPS and production line optimization. More specifically, the workflow of genetic algorithms, simulated annealing, q-learning, cell balancing and deterministic optimization is explained for each of the use cases.
Keywords	Model, genetic algorithm, simulated annealing, q-learning, cell balancing, optimization, MPS

Function	**Name**	**Entity**
Author	Irene Torrego	Accuro
Editors	Irene Torrego	Accuro
Contributors	Ali Kafalı	ACD
	Muhammed Oguz Tas	Inovasyon Muhendislik
	Irene Torrego	Accuro
	Murat Sağlam	Alpata
	Bruno Mota	ISEP
Reviewer		

Executive summary

In order to achieve the goal of rearranging the factory layout, production lines and production planning to optimize profuction efficiency and flexibility, several altorithms and mathematical methods will be used. This deliverable provides a description of the algorithms being developed and use with the purpose of reconfiguring and organizing factory operations and make production more efficient. These descriptions follow a mathematical notation, presenting the operations the algorithms and models need to perform in order to achive the desired result.

Partner contributions record

#	Entity	Contributor on Phase 1	Date of Contribution1	Contributor on Phase 2	Date of Contribution2
1	Accuro	Irene Torrego	10/01/2023		
2	ACD	Ali Kafalı	21/12/2022		
3	Alpata	Murat Sağlam	11/01/2023		
4	Evosoft				
5	Inovasyon	M. Oguz Tas	18/12/2022		
6	ISEP	Bruno Mota	13/01/2023		
7	Progim				
8	SisTrade				

Changes record

Version	Date	Entity	Description of Changes
V0.1	04/11/2022	Accuro	Creation of document template
V0.2	21/12/2022	ACD, Inovasyon	Section 2 first draft.
V0.3	10/01/2023	Accuro	Added information about q-learning applied to production scheduling in §2.5.
V0.4	11/01/2023	Alpata	Updated KPIs for monitoring in §2.7. Added models for master production scheduling in §2.6
V0.5	13/01/2023	ISEP	Added information about the models for production line optimization to minimize total cost and maximize machine longevity in §2.8.
V0.9	30/01/2023	Accuro	First review and formatting.
V0.9.1	31/01/2023	Alpata	
V1.0	31/01/2023	Accuro	Final review and formatting.

Contents

List of figures

List of tables

No table of figures entries found.

1. Introduction

1.1. Document objectives and scope

The objective of this document is to provide information about the mathematical models being used in the project for the configuration of equipment, combining processes from different production lines, transmutation between manual and automated processes, optimisation of the use of resources (machines, human resources, materials, energy, etc.).

There models lead to the algorithms developed in the project. However, this deliverable does not focus on the development of the algorithms themselves or their performance, but on the mathematical underlying principle.

1.2. Document structure

This document is divided in several subsections, each of them providing information for different types of mathematical models depending on their purpose.

2. Mathematical models

In this part, the mathematical models that will be used within the scope of the project for modelling the factory layout and balancing the production systems are explained. These models provide important information for improving production processes and increasing efficiency, which is the aim of the MUWO project.

2.1. Models for reconfiguring the equipment

These models aim to flexibly design production and factory systems using a digital twin-based approach, and to have this design be reconfigurable. To achieve this, multiple 2D – 3D sensors were used to gather up-to-date and instant information about the real production process, and the data from these sensors was synthesized so that the model in the digital twin could be updated instantly. The digital twin offers the opportunity to rearrange the production system and modify it according to the problem by modelling the production systems at different levels, such as assembly processes, production stations, and line levels.

The first model considered in this context aimed to create a production model based on human-factory cooperation [1]. As can be seen from the workflow shown in Figure 1, multiple alternatives have been generated for the locations and layouts of the components. Then, the main workstation was created based on resource planning, and the state of the environment and process was monitored using sensors from there. Since this monitoring is a continuous and instantaneous process, if any anomaly or unexpected behaviour is detected, the system will be aware of this situation, receive the information in the physical world, and allow the digital twin to use this information to rearrange the created workstation and production environment. As previously mentioned, the search algorithms based on artificial intelligence determine how the regulated and reconfigured business environment should be.

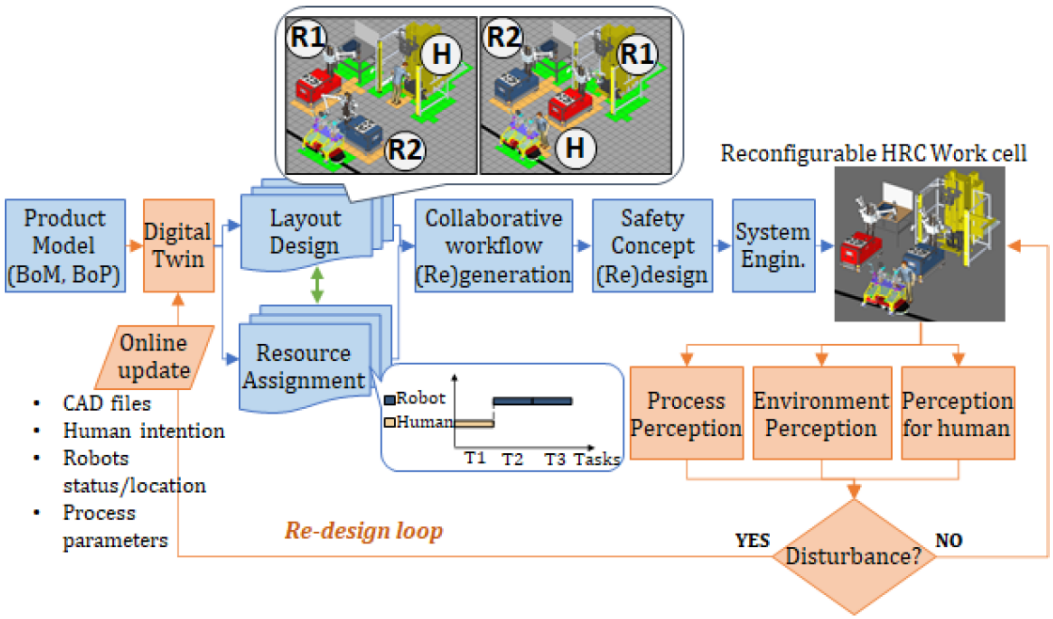

Figure 1. Paradigm on the design workflow of HRC applications.

In another alternative model, an architectural basis for a cloud simulation platform is examined [2]. In this model, a data-based modelling and simulation method was used to achieve automated modelling. Third, the system was implemented in the cloud using Java, MySQL, and the Anylogic platform, and the effectiveness of the proposed method was verified through experiments in a real workshop of a company. In this model, a proposal is made to create a general cloud-based simulation architecture that responds immediately to users' requests and enables the creation of a simulation environment according to these requests. The general outline of the proposed structure can be seen in Figure 2.

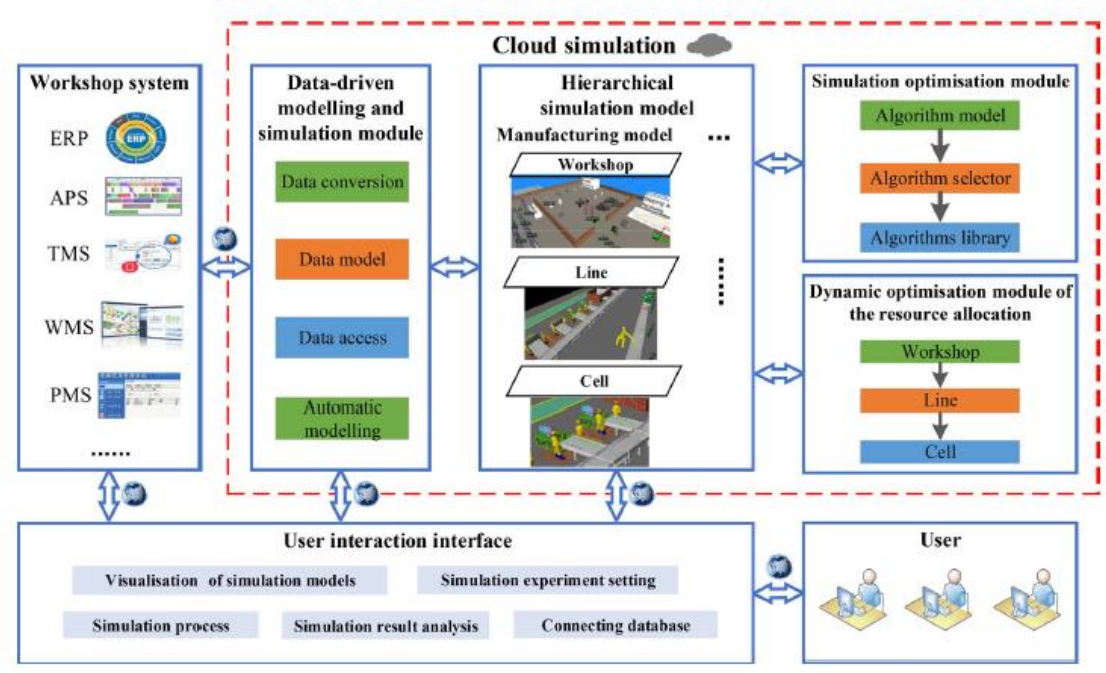

Figure 2. Data-driven cloud simulation architecture for AFPLs in smart factories.

In data-based modelling, simulation models are created using data collected from production environments. With the simulation optimization module and dynamic optimization module, resources are distributed in an optimized manner and optimum planning is created for a flexible and automatic production line. All created modules are placed on the cloud platform, and users can use an interface and simulation model from there. Unlike traditional cloud-based applications, the generalized simulation model is uploaded directly to the cloud. To customize this simulation environment for a specific application or problem, strategy and smart algorithm libraries have been created. This allows the user to customize the simulation environment using these libraries. On the other hand, the desired simulation environment can be created instantly from the data-based general model, which greatly reduces modelling time. In the method, the user is responsible for regularly collecting and updating real-time data. The simulation model is shown in Figure 3.

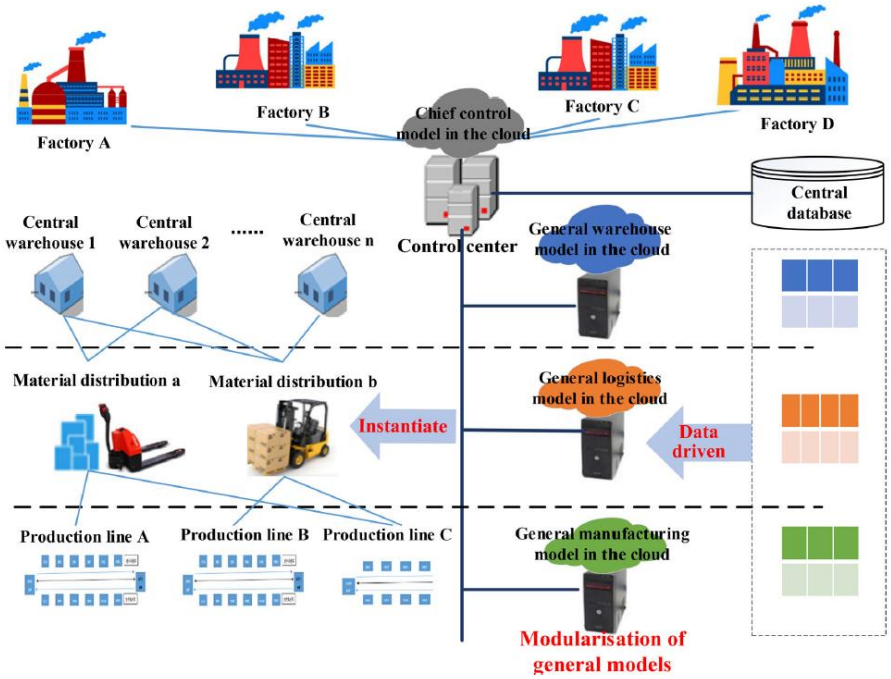

Figure 3. Factory collaborative cloud simulation.

Three steps are required to create simulation models and manufacturing processes:

- First, data is collected and standardized. Then, this data is classified, and a data structure is created. After the data structure is created, the data is divided into three categories: layout data, operation planning, and dynamic data. Layout data sources include the numbers and distributions of sources. It contains operation planning control logic and planning rules. Dynamic data, on the other hand, contains production process information.
- A general library of simulation models is created. A general library of simulation models is created for each different production area and product. This library contains three types of data: production, transportation, and inventory. For example, the production library includes CNC machines and mobile robots. Using these libraries reduces workload and increases efficiency in real-time simulation.
- To automatically create the simulation environment, a source production engine was created, and the layout of the simulation model was provided automatically according to the incoming layout data by using the objects in the general production model library with this created engine. In addition, by looking at the operation planning data, the variables of the resources can be adjusted accordingly. At the same time, these data are updated regularly to ensure that the simulation model moves in coordination with real-time.

In these models, which were created by reconfiguring the factory equipment, making the production processes much more efficient and faster was the main focus.

2.2. Models for combining different production lines

The increasing demand for personalization in products requires high flexibility in production systems to adapt to changes. The open architecture machine tools (OAMT) are a new class of machine tools defined and developed as a fixed standard platform with addable and quickly interchangeable personalized modules. Engineers can flexibly reconfigure the production system to match the process plan by integrating personalized modules into the OAMT. The fundamental enabling techniques are detailed, including how to combine the cyber and physical systems and how to quickly program the production capacity and functionality of production systems to adapt to rapid changes in products. The effectiveness of the proposed approach in reducing the overall costs of the reconfiguration process by automating and optimizing it quickly and in achieving advanced system performance is demonstrated through a physical application. The digital twin-based manufacturing system is a typical representative of smart manufacturing and has several advantages beyond the latest technology. However, when it is necessary to reconfigure a production system to meet new production requirements, manual reconfiguration of the digital twin-based production system is time-consuming and has high labour costs due to the complexity and flawed models of the system. If the production system has industrial robots with complex functions and inflexible programming, this problem becomes even more acute.

The first of the models discussed in this context is about modularizing production systems and rapidly restructuring these systems. This restructuring is important in the manufacture of 3C products (e.g., smartphones) that have fast deadlines and high change rates in the production system. The method aims to realize the production of customer-specific products by increasing the flexibility of a production system. A digital twin-driven approach is proposed to rapidly reconfigure the production system to respond to changes in product orders. The method creates a new reconfiguration through a digital twin-driven rapid restructuring platform after analysing the differences between two products and determining whether the existing system can produce them.

In this sense, a REST (Representational State Transfer)-based IIoT (Industrial Internet of Things) model has been proposed for rapidly reconfiguring the controls and sensor network without human intervention. This approach can quickly switch the production system's capacity and rapidly integrate multiple processes into existing systems, enabling the quick initiation of new product orders. A two-level programming approach of high-level efficiency rebalancing and low-level restructuring cost is offered to find an optimal reengineering solution. The proposed approach can also minimize the overhead of the restructuring process by automating and optimizing it.

As shown in Figure 4, this work used the REST architectural style. In REST, all operational objects are abstracted as resources. The location of resources is identified by a uniform resource identifier (URI). The operation of the resource is determined by the URI and the Uniform Interface (e.g., HTTP). REST provides an architectural principle for building scalable,

simple, portable, and loosely coupled applications. Due to the capabilities of REST, resources are often represented by standard templates. Resources in the cluster can be mapped to the network by creating a REST-based embedded Web server on the smart gateway as the cluster head node. The mapping relationship between a physical resource and a virtual model is defined in mathematical language in a template. Resources are organized in the IIoT network through an intelligent gateway. In a Web network, resources are aggregated according to the physical connection mode of the clustering.

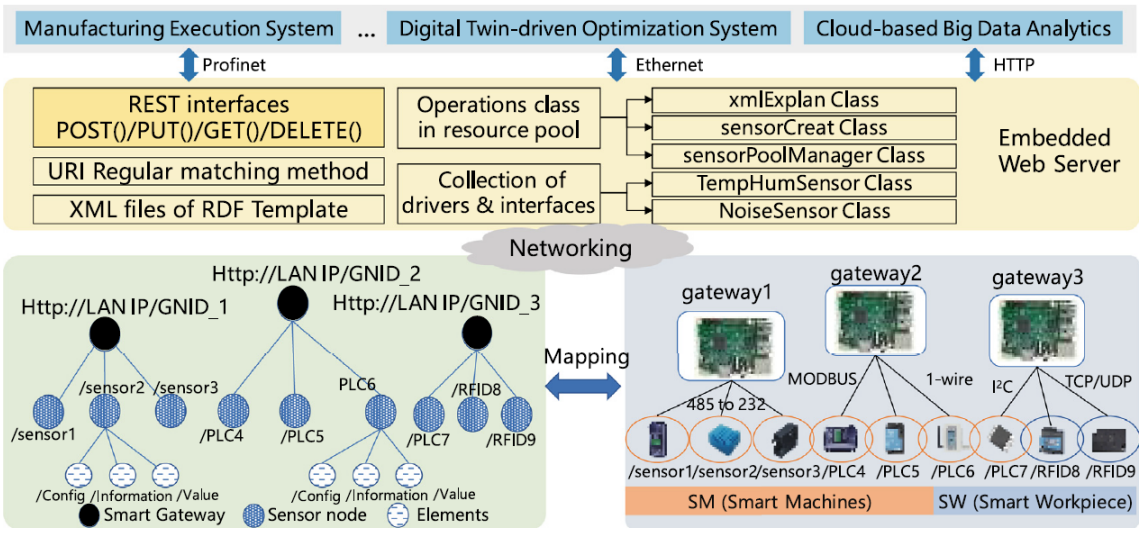

Figure 4. REST-based IIoT system

In the other model digital twin-based manufacturing systems (RDTMS) framework [3] was designed to create digital twin-based manufacturing systems with high accuracy, high applicability, high flexibility, high intelligence, and high reconfiguration capability. The proposed framework uses the five-dimensional fusion model of the digital twin virtual entity (DTVE), which realistically describes the relevant knowledge, capabilities, and rules of the various production resources and environments in the physical space. As shown in Figure 5, the framework of RDTMS is designed to create DT-based manufacturing systems with high accuracy, high applicability, high flexibility, high intelligence, and high reconfiguration capability.

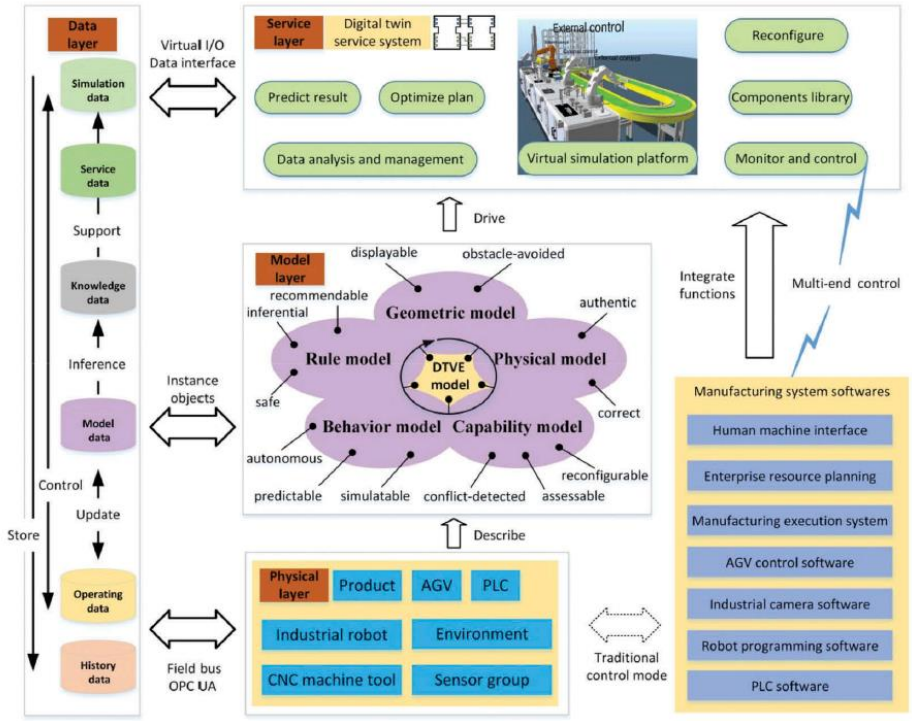

Figure 5. Five-dimensional fusion model-driven framework of RDTMS.

The work mentioned that there are three levels (equipment, unit, and system) in reconfigurable problems of physical production systems (Figure 5). Virtual and augmented reality are used to provide more efficient monitoring and management. The proposed model is a five-dimensional fusion model in the physical production system and is based on DT. The five dimensions of the model are the Geometric Model, the Physical Model, the Ability Model, the Behaviour Model, and the Rule Model.

Geometric Model: An obvious feature of the DT-based production system is that operators can observe in detail the entire scene and production process of the physical production system using virtual and augmented reality, which makes monitoring and management more efficient, intuitive and accurate. Therefore, you can change the shape, size, location, etc. of physical assets. A descriptive Geometric Model is required.

GM = (Shape, Size, Position, Rotation vb.)

Physical Model: One of the requirements of this research is to create a DT-based manufacturing system with high accuracy and high applicability. Therefore, PM is necessary to describe the physical structures, physical rules, and property values of physical entities. It allows virtual assets to simulate the same results as real operations under the same environmental conditions. For example, pallets or other assets in the virtual production system will accelerate and decelerate on the conveyor belt based on the friction and mass value described in the PM. PM also supports several SFBs, such as the SFB of energy consumption calculation.

PM = (Speed, Mass, Friction, Abrasion vb.)

Capability Model: To implement automatic reconfiguration of the DT-based production system, the computer must understand the capabilities and roles of each type of production resource in

the production system; it can also be considered as relevant information to support the SFBs of outcome prediction. Performance evaluation, planning optimization etc. These two opposing capability modes can be abstracted into functional interfaces on an entity. When two entities have a pair of complementary capability interfaces, they will interact in the production system and then the interfaces will be used and transformed into the other two forms of capability i.e. what it does and what it does. Dynamic description of DTVE's capabilities. In addition, the value of a skill and the objectives of applying a skill will also be explained in a particular skill.

CM = (Cando, Isdoing, Canbedone, Isbeingdone, etc.)

Behaviour Model: Every entity in a physical production system has certain operations to perform a complete production process or to implement some function. Virtual entities with manufacturing capabilities, such as industrial robots and computer numerically controlled machine tools, should be treated as self-contained functional units capable of performing a series of autonomous actions under certain conditions (the event tracked in the SFB as shown in Figure 8). It greatly reduces repetitive and inflexible programming. It is also used to implement some services such as automatic running of RDTMS and estimation of production results. Therefore, an UN is needed to abstractly define the behavioral logic of any entity in a physical production system.

BM = (Take, Motoron, Wait, Fault, etc.)

Rule Model: Derivation rules, association rules, constraint rules, etc. There are many rules in the production process and about each asset to support inference and recommendation SFBs that can be divided into. These rules can be derived from big data production. And it can also be defined by experts to make RDTMS safer and smarter. In the reconstruction process, the interface matching rules and spatial calculation rules will be considered by the computer.

RM = (Derivation, Association, Constraint vb.)

To verify the practicality and effectiveness of RDTMS, a robotics-based workshop was used for assembly in this section. An RDTMS prototype was created based on the modeling method proposed in this paper. A system-level reconfigurable task is implemented by the design upgrade strategy and the results are analyzed. In addition, the progress of the modeling approach and the applicability of the reconfigurable problem for DT-based manufacturing systems are explained by comparing it with three other common modeling approaches.

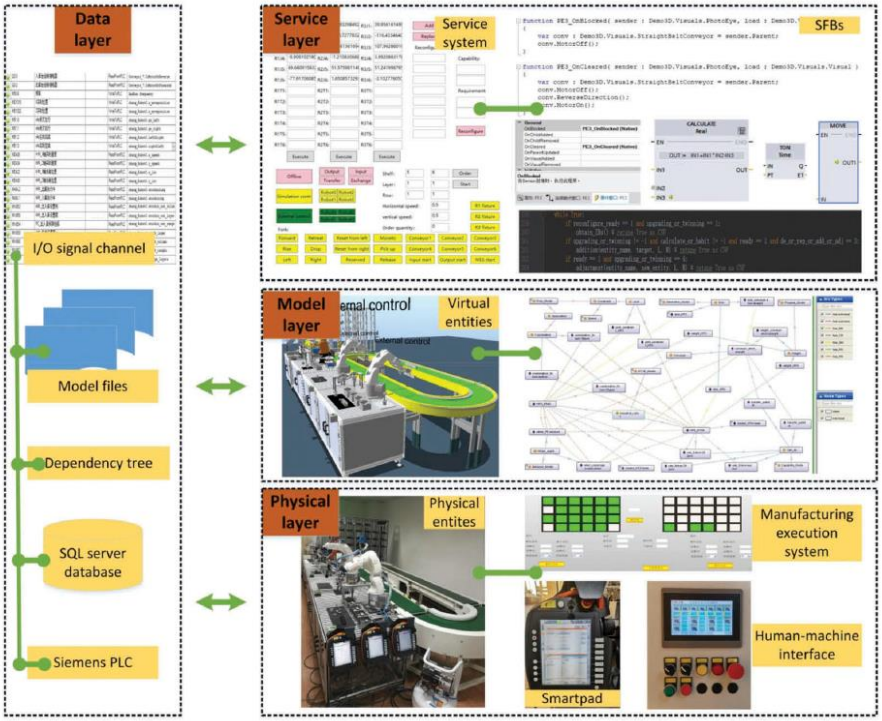

Figure 6.The structure of the prototype system.

The structure of the prototype system is described in Figure 6. Real-time data interaction between physical layer, model and service layers are handled by Siemens PLC, software I/O and SQL server database. DTVE model data, dependency tree and related information data are stored in SQL server database and model files. The control buttons of the equipment and data visualization interfaces are integrated into the virtual simulation platform in the service system of RDTMS.

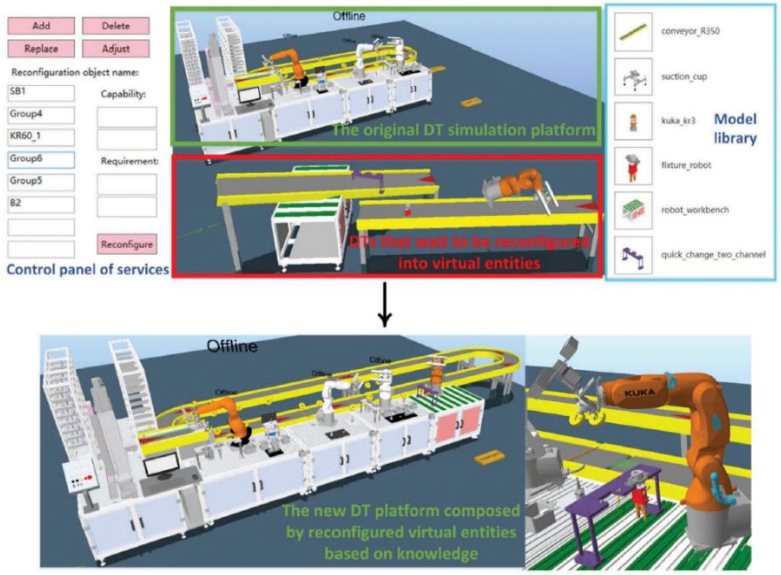

Figure 7.The results of the reconfigurable task

Another approach aims to create flexible digital twins according to changes in design stages. Through the case study, it is aimed to reveal hidden flaws and propose solutions when the digital twin is used in factory design. Thus, it is aimed to reduce the creation time of the digital twin model thanks to the modular approach. However, the fact that different industries have different characteristics and the needs of these industries to use different modules should also be considered. With the designed modular design model, it is planned to facilitate the flexible design of factories. With this model, it has been tried to make it possible for factory designers to design factories by considering instant needs. For example, the production line of a paper cup manufacturing plant in China is modeled. Production routes are as shown in Figures 8 and 9.

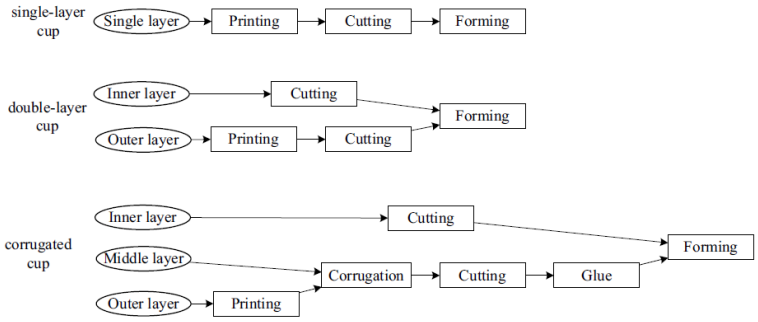

Figure 8. Product routing for a Chinese paper cup factory

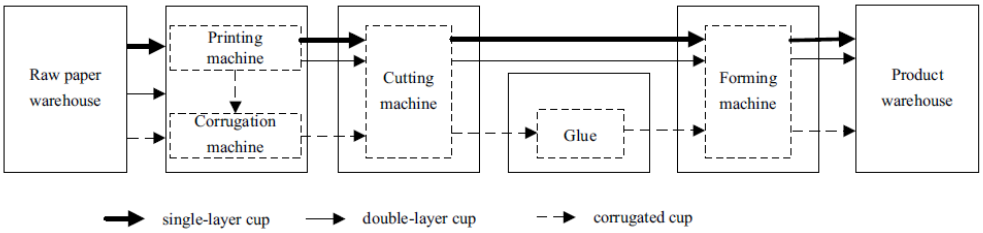

Figure 9. Material flow route between workshops

The construction process and interface of the digital twin is shown in Figure 10. Based on the minimum calculations, the initial layout design can meet the annual output demands and achieve the theoretical equipment utilization rate.

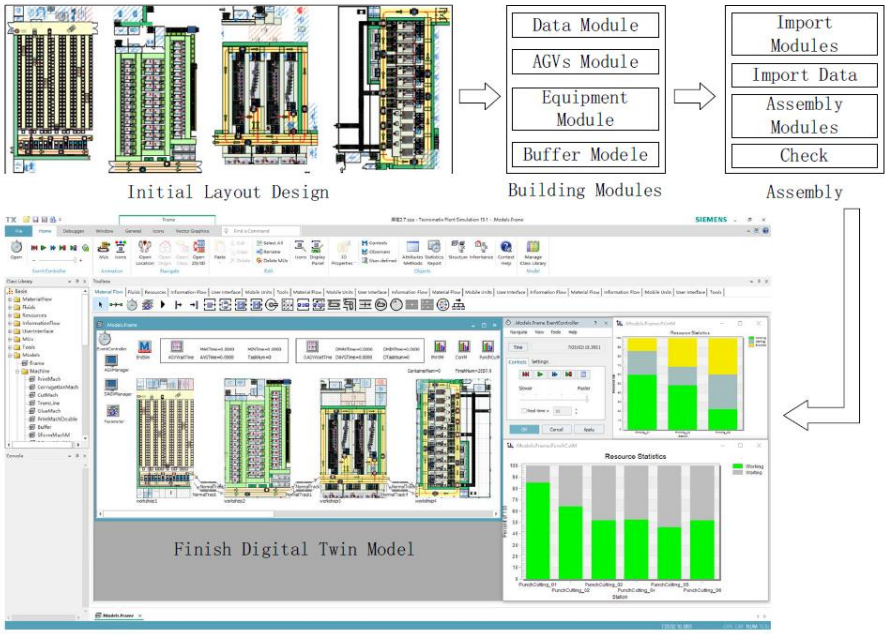

Figure 10. An example of the model building process

2.3. Models for resource optimization

The model [4], which will be exploited for resource optimization, uses digital twin and ALC (Augmented Lagrangian Coordination) method to overcome the problem of credit sensitive resource allocation in shared production.

Shared production is a production model in which the production service is not provided by a single enterprise but is completed by the cooperation between multiple enterprises in different geographical locations. Optimum Distribution of Production Resources in Shared Production aims to coordinate these resources between organizations. The credits of the sourcing providers are considered in order to obtain reliable trading in shared production, as the participants in the shared production (i.e. the sourcing, the customer) may not always know each other. Credit: It includes criteria such as quality evaluation of the manufacturing service, error rate, on-time delivery rate. A digital twin-driven service model has been created to realize remote monitoring and control of shared production resources.

Resource allocation is provided by the Augmented Lagrangian Coordination method. Compared with the Genetic Algorithm to support it, it has been seen that it gives more reliable and more stable results that the source providers take care of their credits. To demonstrate the feasibility of the proposed methods, a platform consisting of three main functions: joint resource management, joint resource allocation and process monitoring has been developed (Figure 11).

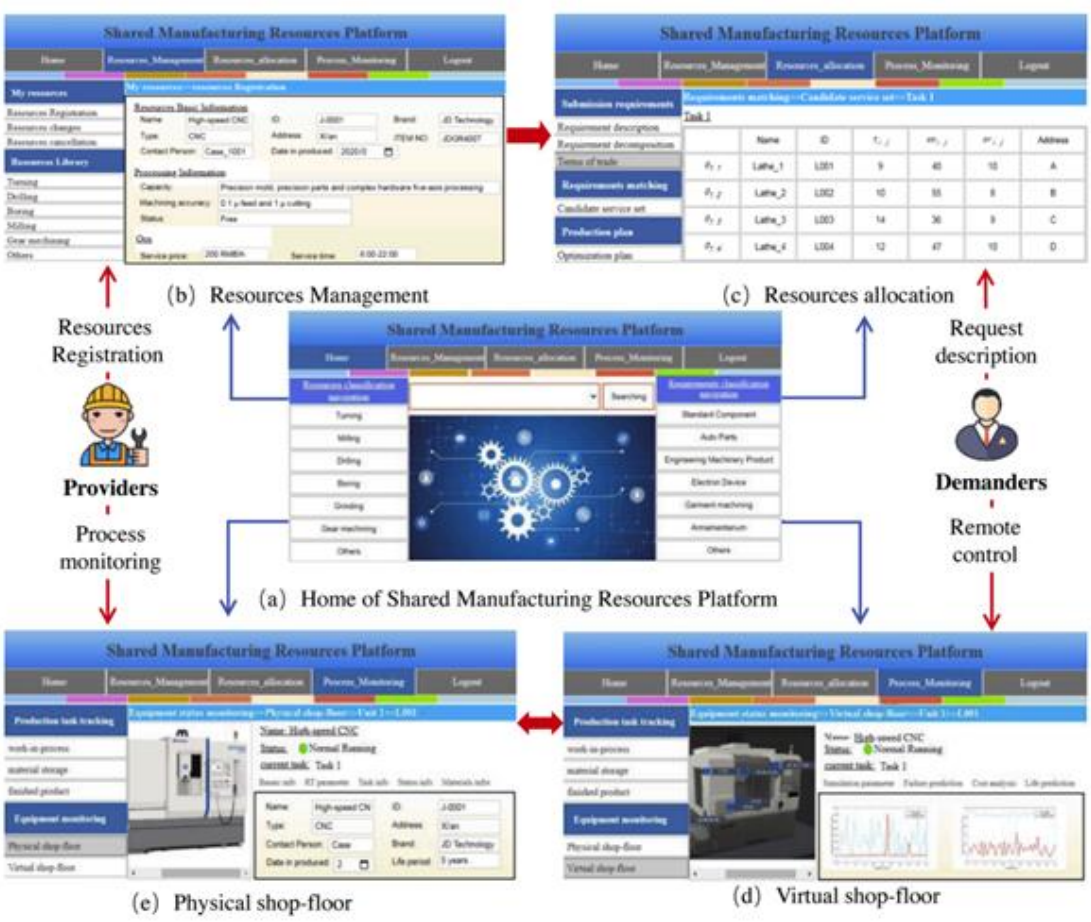

Figure 11. Management platform of shared manufacturing resources.

Figure 11 (a) shows the home page of the platform. It has tabs for resource classification, requirements classification, and resource and requirements. Figure 11 (b) shows the resource management module of the system. Production resources such as provider, machining centers, CNC machine tools and simulation equipment are recorded in a unified and standardized format. The basic information of a CNC machine, processing information and service quality are recorded. Thus, virtual sharing of services has been realized. Figure 11(c) shows the three main steps of the resource allocation module: requirements submission, requirements fulfillment, and production plan. Customers define production tasks uniformly, present their requirements and processing methods. Next, the task is decomposed into subtasks and subtasks are matched to obtain the candidate resource set. Resource allocation is accomplished by calling the optimization method. Finally, a detailed optimized production plan is obtained. Figure 11 (d) and (e) show the production process monitoring module, which consists of a physical production area and a virtual production area. The production monitoring process mainly consists of components such as equipment operation status monitoring, task progress tracking, production progress tracking, material consumption monitoring, process parameter monitoring. Monitoring can be shown in real time. As shown in Figure 13 (d) and (e), the high-speed CNC with ID L001 works to perform task 1 in the physical workshop, simulation and cost analysis are performed in the virtual space.

2.4. Models for smart production

Today, with the development of digital technology, it becomes possible to create smarter, high-performance, and high-efficiency systems by designing the production systems used in the industry in an integrated manner with the developing technologies. While manufacturers want to react quickly and cost-effectively to sudden changes in the global and domestic market, they also want to reach the end user with their safe products and get ahead of the competition in the field, the concepts of "smart manufacturing" or "digital manufacturing" Industry 4.0 brought to the fore thanks to Intelligent production systems; cyber physical system (Cyber Physical System - CPS), internet of things (Internet of Things - IoT), artificial intelligence, big data (Big Data), cloud computing (Cloud Computing), simulation, virtual reality (Virtual Reality - VR), augmented It is a multi-structured system that includes emerging communication and information processing technologies such as reality (Augmented Reality - AR) and digital twin (Digital Twin). Creating an intelligent production system that includes more than one such technology brings problems such as security, system integration and environment independence. It is foreseen that the technologies to be used will have a flexible and reliable network infrastructure with the necessary hardware/software components that can be reconfigured, will make the current system smart and facilitate the use of the necessary technologies.

There are specific strategies for smart manufacturing system transformations. Supply Chain Readiness Level, Manufacturing Enterprise Solutions Association (MESA), Manufacturing Transformation Strategy (based on ISA-95 methods) are common strategies used, but they do not use information and communication technologies as the main method. Using information and processing technologies as the main resource creates smarter business processes and allows the management to operate in an automated way. Combining this smart production process with IoT provides an environment for creating a smart production system by preparing an environment for creating various software, security, artificial intelligence, and analysis methods. They use many technologies in the smart production system;

- Virtual Reality (VR): Helps reduce the cost of prototyping and testing any product. It also helps visualize and test products in a simulated environment for end customers during the digital manufacturing process.
- Augmented Reality (AR): They use technology to combine the real physical environment with computer-generated graphics to visualize components artificially added to an existing real-world scenario for training, simulation, or validation of some production designs before going into actual production. It helps to realize the product in an existing environment. With the demonstration of various conditions in the augmented environment, training and product testing for new employees has been shown to be more efficient and time-saving.
- Cyber-Physical Systems (CBS): The Cyber-Physical System (CPS) collaborates computational entities with the physical world and its ongoing processes using data processing services available directly on the Internet.

- Additive Manufacturing (AM): AM technology is flexible for customization, rapid prototyping, rapid spare parts production, and on-site production, resulting in huge cost savings in production time and machine tool replacement and raw materials, and facilitate reverse engineering of the product.
- Big Data Analytics: Big data analytics helps the manufacturer to identify the current status and causes of product failures in real time, guide customers to buy their products by understanding their purchasing habits and needs, and also teaches the potential of data-driven marketing.
- Flexible and Reconfigurable Manufacturing Systems (FRMS): There are two types of production flexibility. Routine flexibility allows the manufacturing plant to produce new types of products on the same production line, while machine flexibility allows for programming the production routines of different processing stations, sharing processing activity among them. The defining features of FRMS in smart manufacturing are modularity, integrability, flexibility, scalability and diagnostics.
- Artificial Intelligence (AI): Artificial intelligence system, self-decision, self-optimization and automatic response to physical changes such as changing production programs, stopping or starting any machining unit, automatic switching of machine tools and giving timely warning in uncontrollable situations It is used in smart production with the ability to give.
- IoT and IIoT: IIoT focuses on the Industrial application of IoT, which connects every physical asset over the internet. In industries, systems such as IIoT physical sensors and the entire process monitoring and control system are called the internet cloud, which enables each device to interact and collaborate to achieve common goals. It provides the communication infrastructure of human-machine interaction, intelligent process control for welding, tools and material optimization.
- Simulation – Digital Twin: The copy created in the Digital Twin concept, which is defined as a digital copy of a physical object, includes the object's model, data and the ability to track the object. Testing on a Digital Twin created with data from complex products that are costly and difficult to test in real life makes it easy to test the product before releasing it to the physical world. The effect of the data from the tests is processed into the models. Since no physical product is used in the tests, this reduces operating costs and extends the life of equipment and assets. In addition, thanks to the Digital Twin and simulation applications, complex programs are implemented more quickly, thus saving time.

2.5. Models for Scheduling

Scheduling is a decision-making process that seeks to optimize one or more objectives that deal with the allocation of available resources on a time basis to actions that need to be fulfilled. Examples of resources are the machines in a workshop, the actions that need to be performed and the work that needs to be done in the production process. In other words, scheduling is determining when and in what order the workpieces that make up a product will be processed on the machines at hand. With scheduling, problems such as using production

facilities in the most effective way, responding to customer demands as quickly as possible, completing works without delay in delivery dates, shortening the production process, preventing bottlenecks in production, and reducing overtime work are solved. They are two different methods.

2.5.1. Simulated Annealing

Within the scope of the task, it first reads the tasks, the operations of the task, the machine information, the machines suitable for the operations and the processing times of the operations on the machines from the excel format and reads the algorithm related coefficients. Algorithm phases are:

- Initial Solution Creation: The initial solution is randomly generated. Generates a random value for each operation. By sorting these values, it also sorts the operations. Operations are assigned to machines randomly. Thus, it becomes an input to metaheuristic algorithms. The initial solution is ready.
- Searching Neighborhood: The objective function is calculated by considering the initial solution. This value is assigned as a starting point for the current solution and the best solution. As better solutions are obtained in each iteration, the best solution will be updated and improved. Neighborhoods are generated as much as the number of neighbors that need to be looked at from the solution at hand. Randomly 2 operations selected and returns their index. The locations of these operations are interchanged. All operations are then assigned to randomly selected alternative machines. In this way, a new neighborhood is created. Operation sequences and machine operation sequences are corrected. The objective function of this neighborhood is calculated.
- Assess Neighborhood Solutions: If the value of the neighboring solution is better or equal to the current solution (It will be greater or less than the objective function, this will be expressed as better) it is accepted as the current solution and the current solution is updated. If this value is better than the best solution, the best solution is also updated. If the value of the neighboring solution is worse than the current solution, the acceptance probability is calculated. It is accepted or rejected according to the probability of acceptance. The temperature value is used to calculate the acceptance probability. If the probability of acceptance is greater than the random number produced, the solution is accepted and the opportunity for bad solutions is given. Other iterations continue over the accepted neighborhood. If this neighborhood is rejected, the iteration continues by calculating new neighborhoods over the previously accepted neighborhood. After the determined number of neighborhoods are calculated, the temperature value is updated again at the rate of the cooling coefficient before starting a new iteration. When the specified number of iterations are completed, the algorithm stops. The best solution is the result. This solution prints the solution as a table and as a Gantt chart.

2.5.2. Genetic Algorithm

The component first reads the tasks, the operations of the task, the machine information, the machines suitable for the operations and the processing times of the operations on the machines from the excel format, and reads parameter related to algorithm. Algorithm phases are:

- Initial Solution Creation: The initial solution is randomly generated. This generates a random value for each operation. By sorting these values, it also sorts the operations. Operations are assigned to machines randomly. All elements of the population are created in this way. Thus, it becomes an input to metaheuristic algorithms. The initial population is ready. After the population is formed, the objective functions of all the elements are calculated and the best solution is assigned to the current solution and the best solution as a starting point. As better solutions are obtained in each iteration, the best solution will be updated and improved. The following operations are performed for the number of iterations specified at the beginning.
- Selection process: Selection is done by roulette method. In this method, the cumulative probability is calculated. The ratio of the objective function value of each element in the population to the total objective function value becomes the probability of being selected. The cumulative probability of the first element is assumed to be 0 and the cumulative probability of that element is calculated when we add the probability of choosing each element with the probability of choosing the previous element. A random number is derived for each element in the population. Elements of random numbers falling into cumulative probability intervals are selected. In this way, the population is rebuilt.
- Xover Process: The crossover ratio is calculated by multiplying the population number by the crossover probability. Random 2-matched children are selected from the population at the rate of crossover. The crossover points of each 2 children to be crossed are selected. Pairs are crossed at the crossover points. That is, the genes of the first child up to the crossover point are put into the genes of the second child after the crossover point. The genes of the second child after the crossover point are put into the genes of the second child up to the crossover point. The New Population consists of the offspring of the cross. If the number of populations cannot reach the required value due to the crossover ratio, the population number is kept constant by adding the children with the best value according to the objective function to the population. After the crossover, the locations of the repetitive genes and the missing genes are determined in children. Excess genes are replaced by missing genes. Then, the operation order is corrected, and the objective function values are calculated.
- Mutation Process: Mutations are made in each iteration. The mutation rate is calculated by multiplying the mutation rate by the population number and the number of genes. Operations in these genes are assigned to randomly selected alternative machines. After the mutation, the objective function is recalculated.

- Evaluating: The current solution is updated with the best objective function value in the population. If a better solution than the best available solution is reached, the best solution is also updated. As the population is here, the iteration is continued. The algorithm stops when the specified number of iterations is reached. The best solution is the result. This solution prints the solution as a table and as a Gantt chart.

2.5.3. Reinforcement learning

The goal of reinforcement learning is to train an agent to complete a task in an uncertain environment. At each time interval, the agent receives observations, rewards and actions from the environment. The reward is a measure of the success of the previous action (taken from the previous state) with respect to achieving the task goal. Therefore, the agent contains two components: a policy and a learning algorithm.

The policy is a correspondence between the observation of the current environment and a probability distribution of actions to be performed. Within an agent, the policy is implemented by a function approximator with adjustable parameters and a specific approximation model, such as a deep neural network. The learning algorithm continuously updates the policy parameters based on actions, observations and rewards. The goal of the learning algorithm is to find an optimal policy that maximizes the expected long-term cumulative reward received during the task.

Depending on the learning algorithm, an agent maintains one or more parameterized function approximators to train the policy. The approximators can be used in two ways:

- Critic - For a given observation and action, a critic returns the expected discounted value of the cumulative long-term reward.
- Actor - For a given observation, an actor returns as output the action that (often) maximizes the cumulative discounted long-run reward.

Agents that only use critical approximators to select their actions are based on an indirect representation of policy. These agents are also called value-based, and use an approximator to represent a value function (value as a function of observation) or a Q-value function (value as a function of observation and action). In general, these agents work best with discrete action spaces, but can be computationally expensive for continuous action spaces.

Agents that use both an actor and a critic are called actor-critic agents. In these agents, during training, the actor learns the best action it can perform using feedback from the critic (rather than using the reward directly). At the same time, the critic learns the value function from the rewards in order to be able to properly criticize the actor. In general, these agents can handle both discrete and continuous action spaces.

Q-Learning agents

The Q-learning algorithm is a model-free, online, policy-free reinforcement learning method. A Q-learning agent is a value-based reinforcement learning agent that trains a critical approximator to estimate future performance or rewards. For a given observation, the agent

selects and performs the action for which the estimated performance is highest. Q-learning agents can be trained in environments with observation and action spaces, where observation can be continuous or discrete while action must be discrete.

During training, the agent explores the action space using epsilon-greedy exploration. During each control interval, the agent selects a random action with probability ε; otherwise, it selects the action for which the value function is larger with probability $1 - \varepsilon$.

To estimate the value function, a learning agent Q maintains a critic $Q(S, A; \phi)$, which is a function approximator with parameters φ. The critic takes the observation S and the action A as inputs and returns the corresponding expectation of the long-run reward.

For critical approximations, table-based value functions are used, the parameters in ϕ are the actual values of $Q(S, A)$ in the table.

During training, the agent refines the parameter values in ϕ. After training, the parameters remain at their tuned value and the trained value function approximator is stored in the critical approximator $Q(S, A)$.

A deep neural network is used to approximate the Q-value function within the critique. This network must have two inputs: one for observation and one for action. The observation input must accept a four-element vector. The action input must accept a two-element vector. The network output must be a scalar, representing the expected cumulative long-run reward when the agent starts from the given observation and performs the given action.

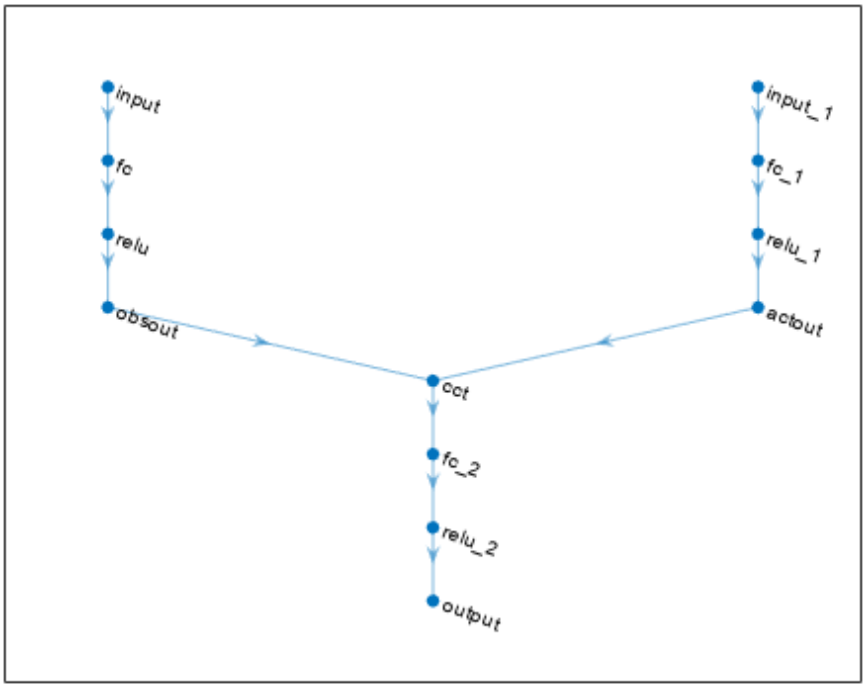

Figure 12. Neural network architecture for the parameter net

Q-learning agents use the following training algorithm.

First, the critical approximator $Q(S, A; \phi)$ is initialised with random parameter values in ϕ. For each training episode, the following process is performed:

1. The initial observation S is obtained from the environment.
2. The following is repeated for each step of the episode until S is a terminal state.
   a) For the current observation S, a random action A with probability ε is selected. Otherwise, the action for which the critical value function is larger is selected.

   $$A = \arg\max_A Q(S, A; \phi)$$

   To specify ε and its decay rate, the Epsilon Greedy Exploration is used.
   b) Action A is executed, observing the reward R and the following observation S'.
   c) If S' is a terminal state, set the objective of the y-value function to R. Otherwise, set it to:

   $$y = \mathrm{R} + \gamma \max_A Q(S', A; \phi)$$

   d) The difference ΔQ between the objective of the value function and the current value of $Q(S, A; \phi)$ is calculated.

   $$\Delta Q = y - Q(S, A; \phi)$$

   e) Update the critical approximator using the learning rate α. Specifying the learning rate when creating the critic by setting the LearnRate within the agent options object.

   For table-based critics, the corresponding $Q(S, A)$ value is updated in the table.

   $$Q(S, A) = Q(S, A; \phi) + \alpha \cdot \Delta Q$$

   For all other types of critics, the gradients $\Delta\phi$ of the loss function with respect to the parameters ϕ are calculated. Then, you update the parameters based on the calculated gradients. In this case, the loss function is the square of ΔQ.
   f) Finally, the observation S is set to S'.

This algorithm will be used in the ALBERO use case (UC3 - Spain) within production scheduling to decide to what extent a machine or an operator is interchangeable for others with similar capabilities to perform a task. In this sense, the theoretical concepts explained above are associated with real elements of manufacturing in UC3 as follows:

- The operator will be the agent in the model.
- The actions will be the work, the route and the time executed by the operator.
- The environment will be the machinery or the function to be executed such as using a tool or transporting material.
- The interpreter will be the algorithm that evaluates the output obtained from the actions and generates a reward for the agent.
- The states correspond to the situation of the assembly line, i.e., the state of the product and the availability of the machines and operators.

Through the Beacons and the data collection interfaces located in the shop floor, the time spent by an operator on the different machines will be obtained and, depending on the production carried out, a reward will be assigned to him. In this way, it will be possible to

determine whether an operator is optimizing his work or whether, on the contrary, he is performing an inefficient job or route.

Operators can then be given guidelines to increase their production and performance.

2.6. Models for Master Production Scheduling

Master production schedule (MPS) is a plan for individual commodities to be produced in each time period such as production, staffing, inventory, etc. It is usually linked to manufacturing where the plan indicates when and how much of each product will be demanded. This plan quantifies significant processes, parts, and other resources in order to optimize production, to identify bottlenecks, and to anticipate needs and completed goods. Since a MPS drives much factory activity, its accuracy and viability dramatically affect profitability. Typical MPSs are created by software with user tweaking.

Set

The set of lines $H = \{1,2,n\}$, line index = $h;$

The set of weeks $W = \{1,2,\dots,r\}$, the week index = w ;

The set of finished products $M = \{1,2,\dots,s\}$, the finished product index = m;

öüp: preliminary production parameter $(0,..,z)$

b: blocking week $(B = 1,..,c)$

Parameters:

- **h**. your line is W. let the $weekly\ working\ hour\ capacity\ =\ t\_hw$ for the week. $w.\ h$ per week. the matrix containing the total working hours of the line will be marked with T: $T = (t\_hw)$;
- **m**. the unit quantity of the finished product is h. even the matrix containing the amounts of hours required for it to be labeled will be marked with A: $A = (a\_mh)$, m when calculating this number. the unit quantity of the finished product is h. even the number of workers (man) – hours required to be labeled can be used;
- **m. W_m** of the finished product. the matrix containing the quantities (requests) that needed to be delivered by the end of the week would be marked with $D = (d\_(mW\_m))$;
- **i_r**: the amount of i content based on the r prescription
- **l**: time limit for blocking
- **〖ym〗_(i_r mw)=w**. The amount of semi-finished products based on the content for the m product at the beginning of the week
- **〖sf〗_(mwi_r)=w**. The $order + forecast$ amount based on the i content of the m product at the beginning of the week

Decision Variables

- $s_{bm} = \begin{cases} \text{if blocking is to be done for 1 m product} \\ 0\ o/w \end{cases}$

- $d_{\ddot{o}\ddot{u}pm} = \begin{cases} \text{if there is a preliminary production parameter for 1 m product} \\ 0\ o/w \end{cases}$
- $v_{mwi} = \begin{cases} 1, \sum_{i_r} sf_{mwi_r} - ym_{i_r mw} \leq 0 \\ 0, o/w \end{cases}$ the amount of semi-finished products based on the content for the m product at the beginning of the week, w. if the i content of the m product at the beginning of the week is greater than the $order + forecast$ amount, it will receive a value of 1.
- $f = \begin{cases} 1, \sum_h a_{mh} x_{mhw} < l \\ 0\ o/w \end{cases}$ if the total production time of the m product per week is less than the blocking limit, $f = 1$.
- $j = \begin{cases} 1, fs_{bm} = 1 \\ 0,\ fs_{bm} = 0\ o/w \end{cases}$ product is made and the justification for that week production time the product is smaller than L, if 1 takes the appropriate value $x_{mhw} = m.$ of the product $w.h$ per week. even the amount produced. $y_{mw} = 1$, if $m.$ manufactured $w.$ if it will be produced per week, = 0 if it will not be produced

Model

$$min\ z = \sum_{h,w} (t_{hw} - \sum_{m} a_{mh} x_{mhw}) \tag{1}$$

$$(1-j) \sum_h \sum_{\ddot{o}\ddot{u}p} x_{mh(w-\ddot{o}\ddot{u}p)} + j \sum_h \sum_b x_{mh(w+b)} \leq (1-j)\mathrm{d}_{\mathrm{mW_m}} + \sum_b j\mathrm{d}_{\mathrm{mW_{m+b}}} \quad \text{(h=1,..,n)} \tag{2}$$

$$\sum_m a_{mh} x_{mhw} \leq t_{hw},\ \forall\ h,\ \forall\ w \tag{3}$$

$$v_{mwi} \leq \sum_h x_{mhw} \leq (1 - v_{mwi}) \max (sf_{mwi_r} - ym_{i_r mw})\mathrm{d}_{\mathrm{mW_m}} + v_{mwi}\mathrm{d}_{\mathrm{mW_m}}, \quad \forall\ m, \quad \forall\ w, \forall\ i \tag{4}$$

$$y_{mw} \leq x_{mhw} \leq M y_{mw} \tag{5}$$

$$x_{mhw} \geq 0 \tag{6}$$

(1) The objective function expresses the remaining free time by subtracting the total duration of the products produced on that line from the total weekly production time on a line. Then, by summing this function from the line and week basis, it is aimed to find all the

free time. By minimizing the objective function, the objective is to minimize the total free time.

(2) The purpose of the restriction is to produce a product by ignoring the preliminary production parameter if a product can be blocked and the weekly production time of the product is less than the maximum time required for the blocking Decommissioning. Otherwise, if necessary, production will be carried out by considering the preliminary production parameter.

(3) It prevents the total duration of the products to be produced on the weekly line from exceeding the weekly line-based production time.

(4) W. The m product produced per week will be produced as much as the order + forecast if the semi-finished product is sufficient, and if it is insufficient, the number of semi-finished products that can be produced will be produced.

(5) w of the product m. In order for the production to be made per week, the relevant product is defined that week

2.7. Models for KPI Monitoring

The aim is to digitize the process on the shop floor and to better visualize the software development process. Users can define their own KPI in the system, select from related KPIs and open them easily on a monitoring screen. Related to the formulas of KPIs or meaningful criteria for the KPI can be selected. A general monitoring screen will be designed according to KPI groups. On this screen, the current KPI value of the facility, the KPI value of the last week and the last month will be displayed numerically and graphically.

In addition to the KPIs defined in the system, the user can add his own KPI. While defining the KPI, the name of the KPI is written. Level selection is made. There are 3 levels: tactical, operational and strategic. The unit of the calculated value of the KPI is selected. The user can select the unit they want to see differently. The KPI trend is selected as either ascending or descending. The measurement frequency should be specified to indicate how often the KPI will be measured. The KPI source must be selected. The value ranges of the KPI should be defined. Significant ranges of the KPI should be specified. There are 3 ranges of values, minimum, normal and maximum. Warning messages to be displayed in the system when values are received in these intervals are determined according to this interval. When it falls below the minimum value, when it is between the minimum value and the normal value, when it is above the normal but below the maximum value, and when it exceeds the maximum value, 4 messages are entered into the system.

A meaningful formula is specified for anyone reading it in the KPI description. In the formula field, the formula to be calculated by the system should be written by selecting the appropriate parameter and dragging it. It is necessary for the proper creation of the query in order to pull the correct data from the database. The KPI resource is selected. By using this KPI value,

the area to be improved is entered. Necessary notes are written. When calculating the KPI, they are selected from which tables are fed data.

Monitoring screens of KPIs will be derived from the same form and different KPI monitoring screens will be created and added to the menu. The menu name is entered, one or more KPIs to be followed in the form are selected, the appropriate filters for this KPI are selected and saved.

On the KPI monitoring screen, the date selection can be selected on a daily, weekly and monthly basis. Increases and decreases can be followed by visualizing them with graphics. Data is also presented as a table. Appropriate criteria can be listed for filtering the data. These criteria should also filter each other when necessary. For example, when a fault type is selected, only faults belonging to this fault type are listed.

On the KPI general monitoring screen, the current value of the most important KPIs selected according to the KPI levels, the last 1 week, month or year value and a general graphic screen are displayed.

KPI definitions are made through the interface and some KPIs are given in the table below.

KPI Name	Note	Unit	Frequency	Explanation	Range
OEERate	Ratio of Fully Productive Time to Planned Production Time	Ratio	Day/Week/ Month	(∑ Fully Productive Time) / (∑ Planned Production Time) * 100	0-100-Ratio Range
PerformanceRate	Ratio of Net Shift Time to Working Time	Ratio	Day/Week/ Month	(∑ Net Shift Time) / (∑ Working Time) * 100	0-100-Ratio Range
QualityRate	Ratio of Fully Productive Time to Net Working Time	Ratio	Day/Week/ Month	(∑ Fully Efficient Time) / (∑ Net Working Time) * 100	0-100-Ratio Range
Mean time between failure (MTBF)	Mean time between failures	Minute	Day/Week/ Month	(∑ uptime) / (Total number of failures)	2-50-Minute Intervals
Mean time to repair (MTTR)	Average time between repairs	Minute	Day/Week/ Month	(∑ repair time) / (Total number of failures)	0-100-Ratio Range

KPI Name	Note	Unit	Frequency	Explanation	Range
Percentage reduction in defect rates	Error rates	Piece	Day/Week/ Month	(∑ number of defective products) / (Total number of products) *100	0-100-Ratio Range
PPM	Parts per million	Piece	Day/Week/ Month	(∑ Number of Scrap) / (Net Production) * 1000000	0-10000-Product Quantity

2.8. Models for Production Line Optimization to Minimize Total Cost and Maximize Machine Longevity

The production line scheduling system for total cost and machine longevity optimization, developed by ISEP, is divided into three main components: data processing characterized by the cell balancing; genetic algorithm represented by the initial population, crossover, mutation, and selection; and finally, the deterministic optimizations defined by the cost and shift optimizations. Figure 13 represents the flowchart of the ISEP production line scheduling system for total cost and machine longevity optimization.

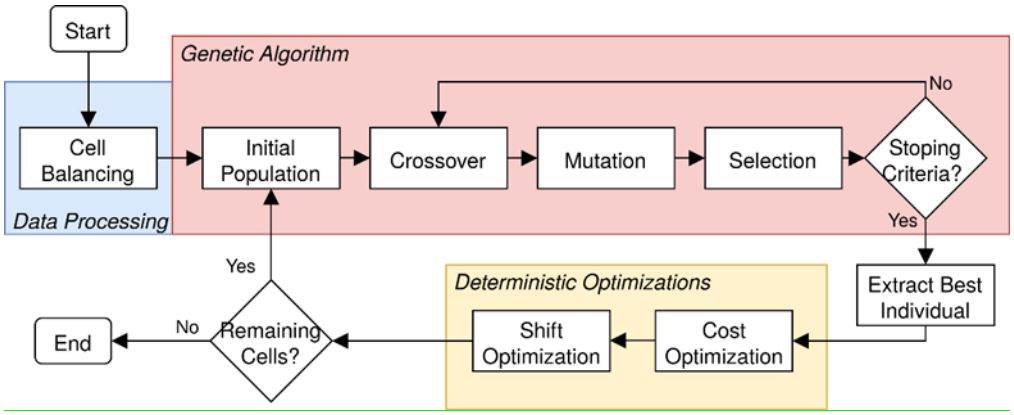

Figure 13. Flowchart of the ISEP production line scheduling system for total cost and machine longevity optimization.

2.8.1. Cell Balancing

The proposed algorithm starts by balancing the requests by the different cells available for manufacturing. This phase begins with the creation of a product request list, sorted by descending order of energy consumption needed to complete a product request. Because the tasks needed to complete a product request can be executed using different task modes with different energy profiles, an average is made considering all the possible task modes. An average of all the possible task modes is considered, above any other type of measure, since it better represents the overall processing time that a task can take.

Afterward, following the product request list, each request is assigned to the cell that has the lowest overall energy consumption, being the product request consumption added to the cell's overall energy consumption. Moreover, the cell assignment process also takes into account if a cell is able to produce the requested product and if it has enough periods to execute the product tasks.

Finally, after each request is assigned to a cell, each cell is executed by a genetic algorithm, followed by deterministic-based optimizations. However, the cell optimizations are connected among them by the general data and comply with general constraints, such as the availability of renewable energy sources that are shared among cells, and energy limit constraints that are applied to the plant floor (i.e., the sum of cells).

The cell balancing optimization focuses on maximizing energy distribution among cells, thus allowing more flexible request management, which in turn reduces the overall energy cost.

2.8.2. Initial Population

The ISEP Genetic Algorithm (GA) starts by initially creating an initial random population, respecting all the imposed constraints. To facilitate the creation of an individual and decrease the rate of not feasible individuals, the algorithm takes into account the compatible machines for each task and prioritizes tasks with a lower number of compatible machines and with a higher processing time. It is noteworthy that, in this stage of the algorithm, maintenance activities and machine setups are treated as tasks with imposed constraints, such as when a maintenance activity could begin and end, and that setups must precede a certain task. After knowing the list of tasks associated with each machine, a two-dimensional matrix is created, which represents the work plan of the job shop, where each line represents the plan of a machine, and each column represents a period. Figure 14 shows an example of such a matrix, where tasks, which some of them can represent a maintenance activity or a machine setup, are identified by colors and their identifiers.

Machine/Period	1	2	3	4	5	6	7	8	9	10
Machine 1	t1	t1	t1	t7	t7	t8	t8		t9	
Machine 2		t5	t5				t2	t2	t2	t2
Machine 3	t4	t6	t6	t6	t6	t6	t6	t3	t3	t3

Figure 14. Example of an individual matrix (machine/period), from the ISEP genetic algorithm, where tasks are identified by colors and their identifiers.

After an individual matrix is generated and at least one constraint is not respected, there are implemented methods that try to repair the generated individual (i.e., by shifting tasks left or right, swapping tasks). In case the repair is not successful, another individual is generated.

Since the number of columns (i.e., time period) corresponds to the time window of the schedule, every matrix from each cell will have the same number of columns but may differ in the number of rows, as cells can have a different number of machines associated.

2.8.3. Crossover

A new generation begins with the crossover of the population of the previous generation. The crossover performed is done between two individuals randomly chosen, which have not yet been crossed; this is guaranteed through a permutation of the population before the crossover is done. The crossover is a two-dimensional type, which will try to balance the tasks coming from each parent (individual). The crossover between two individuals (parents) begins with the formation of the list of tasks for the entire plan, ordered in decreasing order of processing time, and if they have equal times, it is ordered in ascending order of the number of machines compatible with the task. This, therefore, allows reducing the rate of invalid crosses between two individuals. In order of the task list, each task is taken and inserted in the child according to the parents' coordinates; first, it starts with parent 1, then it changes to parent 2, then parent 1 again, and in general, follows this sequence. Another child is made from the same parents but starts with parent 2 coordinates. Figure 15a represents the first four steps in a crossover, starting from parent 1.

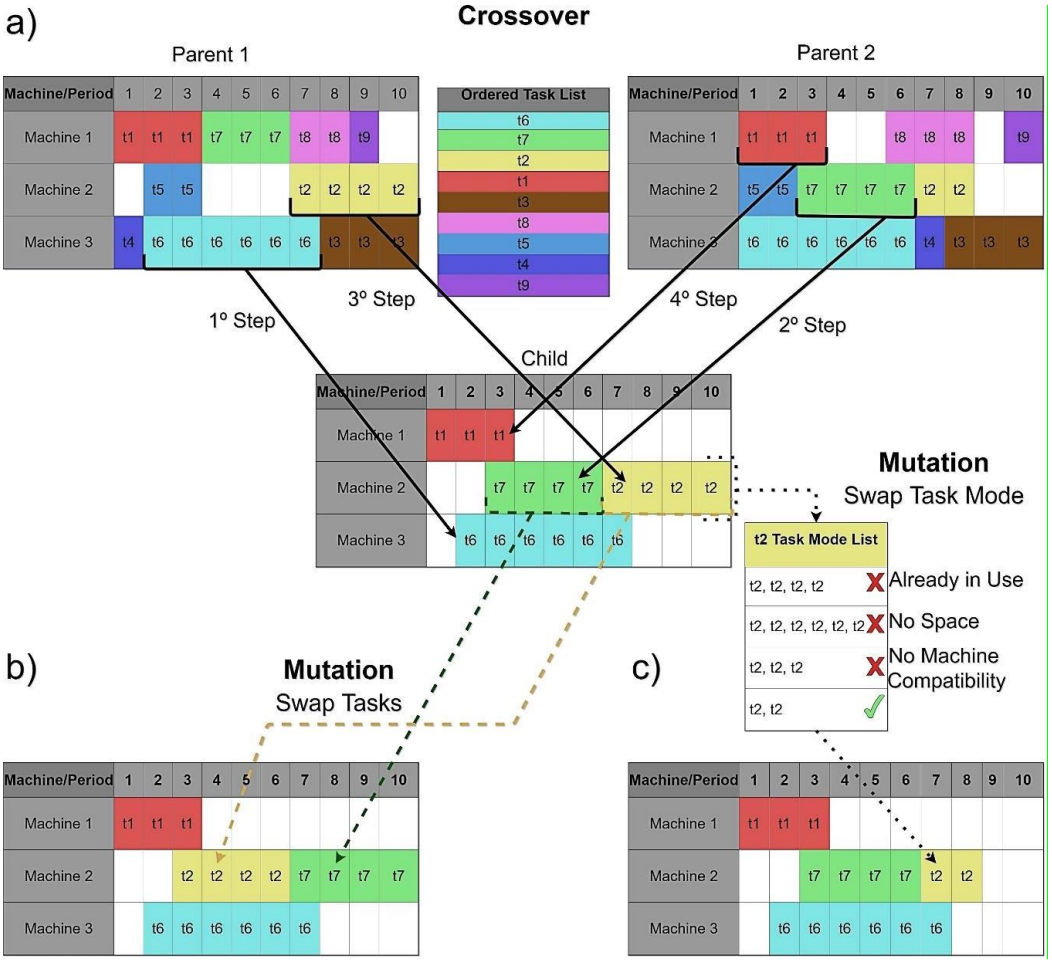

Figure 15. Individuals' crossover and mutation from the ISEP genetic algorithm. (a) Example of the first 4 steps in a crossover, starting from parent 1; (b) example of a swapping tasks mutation; (c) example of a swapping the task mode on a task mutation.

After each crossover, where there are no incompatibilities between parent 1 coordinates and parent 2 coordinates, the resulted child is evaluated in terms of respecting all the imposed constraints. If at least one of the imposed constraints is not respected, the crossover is considered invalid.

In case of an invalid crossover, a perfect copy of either parent 1 or parent 2 is done to the child. This is done since afterward, in the selection phase, all the repeating individuals are eliminated, thus removing the child from the population pool. In short, if an invalid crossover occurs the child is never added to the population.

The most common approach to a crossover in a matrix is separating the parents based on a cutting point(s), in either a column or row manner, in other words, a probabilistic strategy. However, due to the problem at hand, many problems would arise such as task splitting, a high overlap of tasks, between the parents, and inconsistencies (i.e., tasks can have different processing times on each parent, due to task modes). Therefore, it was necessary to use a more deterministic approach to solve these issues.

2.8.4. Mutation

Using the population obtained from the crossover, a mutation is performed. In this procedure, the algorithm takes each individual, and according to the percentage of mutation defined in the input data, it decides whether or not the mutation will be applied to a given individual. However, if the mutation is applied, one of three different types, using equal probabilities, is randomly chosen:

- Swapping tasks, changing two tasks in order of execution and/or machine. Figure 15b represents an example of a swapping tasks mutation;
- Swapping the task mode on a task, affecting energy consumption and processing time. Figure 15c represents an example of swapping the task mode on a task mutation;
- The combination of both swapping tasks and task mode.

If a mutation leads to an invalid individual (i.e., not respecting a certain constraint) then the mutation is reversed and another mutation, affecting different tasks in the schedule, is tried. Nevertheless, if no valid individual can be reached with a mutation, after a short number of tries, no mutation is done.

2.8.5. Selection

The selection phase begins with the union of the new and the old populations, that is, the crossed and mutated population with the initial population of the previous generation. Additionally, repetitions of individuals are eliminated.

Afterward, each individual is evaluated according to their fitness score, which follows a multi-objective function that minimizes the overall total costs, maximizes profit from selling energy, and minimizes machine occupancy deviation.

The first two objectives (i.e., minimize costs and maximize profit) can be joined into a single function which is divided into four fundamental equations: period energy consumption, period energy to pay, period maintenance to pay, and total cost.

The Period Energy Consumption (PEC), represented by $PEC_{Demand(p)}$, gives the total energy consumed by the tasks in a given period p, it can be described by eq. (7).

$$PEC_{Demand(p)} = \sum_{m=1}^{M} E_{Demand(p,m)} \times P_{Machine(p,m)} \quad (7)$$

The variable p portrays a specific period, m describes a machine index, and M the total number of available machines for production. Variables p and m can be compared to the x (i.e., column) and y (i.e., row) cartesian coordinates, respectively, in order to navigate in the individual matrix. The energy consumption of a machine m in period p is described by $E_{Demand(p,m)}$. Furthermore, if a machine priority constraint is applied, variable $P_{Machine(p,m)}$ represents the priority of machine m in period p. For $P_{Machine}$ values above 1, the

priority is decreased, while below 1 it is increased, as a result, neutral priority (i.e., no priority associated) is represented by the value 1.

Regarding the Period Energy to Pay (PEP), portrayed as $PEP_{Demand(p)}$, it represents the energy to pay (i.e., energy cost) in a given period p, it is represented by eq. (8).

$$PEP_{Demand(p)} = \begin{cases} PEP_{Demand(p)} = 0, & if\ E_{Generation(p)} = PEC_{Demand(p)} \\ PEP_{Demand(p)} = \left(E_{Generation(p)} - PEC_{Demand(p)}\right) \times E_{Selling\ Price(p)}, & if\ E_{Generation(p)} > PEC_{Demand(p)} \\ PEP_{Demand(p)} = \left(PEC_{Demand(p)} - E_{Generation(p)}\right) \times E_{Buying\ Price(p)} \end{cases} \quad (8)$$

Variable $E_{Generation(p)}$ portrays available locally generated energy that is free of charge (e.g., PV generation) in period p, $E_{Selling\ Price(p)}$ describes the price for selling energy in period p, and $E_{Buying\ Price(p)}$ represents the price for buying energy in period p. In case PEP_{Demand} results in a positive value (i.e., above zero), it indicates that there are energy costs to be paid, while negative values (i.e., below zero) indicate that profit was made by selling generated energy in excess (i.e., all the PEC_{Demand} was covered by $E_{Generation}$ and the rest sold to energy buyers). Subsequently, zero indicates that there are no energy costs to be paid and no profit was obtained from generated energy in excess.

To calculate the maintenance costs, it is used the Period Maintenance to Pay (PMP), represented by $PMP_{Maintenance(p)}$ it portrays the maintenance costs to pay in a given period p, it is represented by eq. (9).

$$PMP_{Maintenance(p)} = \begin{cases} PMP_{Maintenance(p)} = 0, & if\ there\ is\ no\ maintenace\ scheduled \\ PMP_{Maintenance(p)} = M_{In\ Hours\ Price(p)}, & if\ there\ is\ a\ maintenace\ scheduled\ in\ maintenance\ hours \\ PMP_{Maintenance(p)} = M_{Out\ Hours\ Price(p)} \end{cases} \quad (9)$$

Maintenances can be scheduled either in maintenance hours (i.e., the interval of periods in which the maintenance must/can be done) or out of maintenance hours (i.e., not in the stipuled interval of periods, normally has a monetary penalty). Accordingly, variable $M_{In\ Hours\ Price(p)}$ describes the price of a maintenance activity done in maintenance hours in period p, while $M_{Out\ Hours\ Price(p)}$ represents the maintenance price of a maintenance activity done out of maintenance hours in period p.

Finally, the Total Cost (TC) of an individual can be obtained through eq. (10).

$$TC = \sum_{p=1}^{P} PEP_{Demand(p)} + \left(\sum_{m=1}^{M} PMP_{Maintenance(p)}\right) \quad (10)$$

The total number of available periods in the time window of the schedule is represented by the variable P. Also, eq. (10) can be seen as the sum of the energy cost of each individual, determined as a result of the energy balance ($consumption - generation$) multiplied by the respective energy price, and the maintenances cost according to their respective maintenance hours price.

It is noteworthy that, variable $PEP_{Demand(p)}$ already includes the energy costs from all the machines in a given period p. However, variable $PMP_{Maintenance(p)}$ does not include all the maintenance costs from period p, thus the need to incorporate, in eq. (10), the sum of all maintenance costs from all the machines in period p.

To minimize machine occupancy deviation (i.e., machine occupation rates standard deviation), and thus maximize machine longevity by reducing overload and usage of single machines, it is employed a function that can be divided into the following three equations: machine degradation classifier, machine occupation rate, and occupation standard deviation.

To calculate the machine occupancy deviation, only tasks and setups are considered to influence the degradation of a machine, since they require the machine to be working. Therefore, the classification of factors that contribute to the degradation of a machine is done using the Machine Degradation Classifier (MDC), represented in eq. (11) as $MDC_{Factor(p,m)}$, which classifies a factor contributing to the degradation of a machine m in period p with the value 1, otherwise, it classifies it with 0.

$$MDC_{Factor(p,m)} = \begin{cases} MDC_{Factor(p,m)} = 1, & if\ there\ is\ a\ task\ or\ setup\ scheduled \\ MDC_{Factor(p,m)} = 0 \end{cases} \quad (11)$$

The Machine Occupation Rate (MOR), portrayed by $MOR_{Factors(m)}$, gives the occupation rate of factors that contribute to the degradation in a given machine m, it can be described by eq. (12).

$$MOR_{Factors(m)} = \frac{\sum_{p=1}^{P} MDC_{Factor(p,m)}}{P} \quad (12)$$

Lastly, the Occupation Standard Deviation (OSD) of an individual can be determined by calculating the population standard deviation (i.e., not the sample standard deviation), as represented in eq. (13).

$$OSD = \sqrt{\frac{\sum_{m=1}^{M}\left(MOR_{Factors(m)} - \left(\frac{\sum_{m=1}^{M} MOR_{Factors(m)}}{M}\right)^2\right)}{M}} \quad (13)$$

After obtaining both the TC and OSD for each individual a Min-Max normalization approach is taken, using the results obtained from the individuals in the population, to normalize the TC

and OSD values of each individual in the population. Then, each individual is evaluated according to the Fitness Score (FS), described by eq. (14).

$$FS = TC_{Norm} \times W_{TC} + OSD_{Norm} \times W_{OSD} \quad (14)$$

Variables TC_{Norm} and OSD_{Norm} describe the normalized TC and OSD values, respectively, of an individual. The optimization weights, defined by the user in the input data, for the overall costs (i.e., TC) and machine occupancy deviation (i.e., OSD) are represented by variables W_{TC} and W_{OSD}, respectively. The sum of variables W_{TC} and W_{OSD} must always be 1, and they must assume a value from 0 to 1, inclusive.

The selection of the n best individuals is made according to the input parameter of the algorithm, chosen by the user. The remaining individuals (i.e., population size less n) are obtained from non-elite tournaments. Each tournament consists of two individuals randomly chosen, where they compete based on their fitness scores (i.e., FS). The algorithm calculates the chance of individual 1 winning the tournament using eq. (15).

$$Individual^{1}_{chance} = 1 - \frac{fit1}{fit1 + fit2} \quad (15)$$

where $fit1$ and $fit2$ represent the fitness of individual 1 and individual 2, respectively. Then, a random decimal number between 0 and 1 is generated. If the generated decimal is lower than the chance of individual 1 winning, eq. (15), then individual 1 is declared the winner. Otherwise, individual 2 leaves victorious. Therefore, the individual with the lowest fitness, which in turn has the lowest combination of overall cost and machine occupancy deviation, is the one most likely to be chosen.

2.8.6. Extract Best Individual

At the end of the selection phase, the next generation begins with the population obtained from this selection, and all the procedures mentioned above are repeated. Finally, after a stop condition is met, the best individual is extracted from the last population of the genetic, that is, the lowest balance of cost and machine occupancy deviation job shop schedule, generated by the ISEP GA. Thus, in general, the ISEP GA, through eq. (14), which represents the multi-objective function, tries to minimize the overall cost and machine occupancy deviation of the elaborated job shop schedule as much as possible, taking into consideration constraints, such as task orders, task collisions, energy limits, request deadlines, PV generation, etc.

2.8.7. Cost Optimization

After the execution of the ISEP GA, a post-processing phase is required to adjust/further optimize the result given by the ISEP GA through a deterministic approach.

The cost optimization function will analyze the result of the ISEP GA, and it will identify periods with lower energy costs that do not have an assigned task, then it will try to place the tasks with higher energy consumption to the lower energy cost time slots. If tasks can be exchanged while resulting in a cost reduction and respecting all the constraints, the exchange is carried out. This optimization function uses the energy cost resulted from the prices of all energy sources available, including local renewable energy sources.

It is worth noting that this deterministic optimization only affects tasks scheduled in the same machine, and as such, tasks are never swamped between machines. Therefore, this optimization does not interfere with the machine occupancy deviation optimization done by the ISEP GA.

2.8.8. Shift Optimization

Shift optimization is made by sliding tasks and/or maintenance activities in time to reduce the time between them. This optimization differs from the cost optimization because no task/maintenance will be changed, only slide to near periods. Its main purpose is to join tasks and maintenances, reducing the time between them, and increase space for future unexpected product requests and maintenance activities. For each isolated task/maintenance (i.e., with empty periods on the left or on the right), the shift optimization will try to shift it to the left, and if all constraints are valid and the overall costs decrease or are maintained within the shift margin constraint level, the shift is performed. If the left shifting is not possible, then the shift optimization tries to shift the task/maintenance to the right, applying the same conditions described before. Figure 16 shows an example of the shift optimization performed to a machine work plan, from the ISEP production line scheduling system.

This optimization also only swaps scheduled tasks/maintenances in the same machine, thus it does not interfere with the machine occupancy deviation optimization done by the ISEP GA.

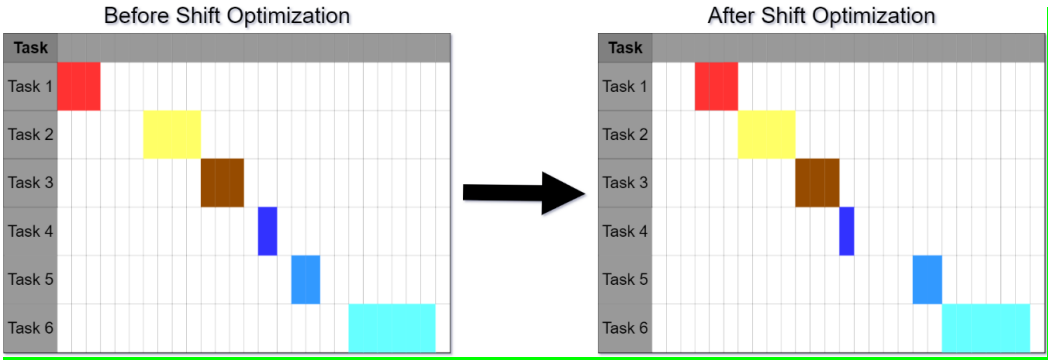

Figure 16. Example of the shift optimization for a machine work plan, from the ISEP production line scheduling system.

3. Conclusions

With the information presented in the previous sections, we can conclude that there is no single model to achieve the objectives set out in the project, but that there are different ways of doing so. In fact, different methods are used in the different use cases, for example, to plan production.

While in UC2 (Turkey) two mathematical models - a genetic algorithm and a simulated annealing model - have been used to read the tasks to be performed, determine which machine is the most suitable to perform them, and combine the results of the two algorithms based on the results obtained in the simulations, in UC3 (Spain) this will be done with a Q-learning algorithm that will determine the optimal machine - operator - priority configuration to determine how a task is to be performed. However, UC1 (Portugal) proposes the combination of cell balancing, a genetic algorithm and deterministic optimizations to maximize the energy distribution in the cells, determine the task distribution and optimize the results both in terms of cost and task distribution. In addition, UC1 presents a different perspective as to which parameters are taken into account to optimize production, the priority here being energy, costs and time spent on both production and maintenance tasks.

Although it may seem counterproductive that in each use case a different combination of algorithms is used despite pursuing the same objective, this plans the possibility of being able to compare which combination provides better results or is more suitable depending on the type of production involved.

4. References

[1] Kousi, N., Gkournelos, C., Aivaliotis, S., Lotsaris, K., Bavelos, A. C., Baris, P., ... & Makris, S. (2021). Digital twin for designing and reconfiguring human–robot collaborative assembly lines. Applied Sciences, 11(10), 4620.

[2] Luo, D., Guan, Z., He, C., Gong, Y., & Yue, L. (2022). Data-driven cloud simulation architecture for automated flexible production lines: application in real smart factories. International Journal of Production Research, 60(12), 3751-3773.

[3] Leng, J., Liu, Q., Ye, S., Jing, J., Wang, Y., Zhang, C., ... & Chen, X. (2020). Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model. Robotics and Computer-Integrated Manufacturing, 63, 101895.

[4] Zhang, C., Xu, W., Liu, J., Liu, Z., Zhou, Z., & Pham, D. T. (2021). Digital twin-enabled reconfigurable modeling for smart manufacturing systems. International Journal of Computer Integrated Manufacturing, 34(7-8), 709-733.

[5] Wang, Gang, et al. "Digital twin-driven service model and optimal allocation of manufacturing resources in shared manufacturing." Journal of Manufacturing Systems 59 (2021): 165-179.

[6] Phuyal, S., Bista, D., & Bista, R. (2020). Challenges, opportunities and future directions of smart manufacturing: a state of art review. Sustainable Futures, 2, 100023.