SOON: Social Network of Machines to Optimize Task Scheduling in Smart Manufacturing

Hatem Ghorbel
Haute École Arc Ingénierie HES-SO
St-Imier, Switzerland
hatem.ghorbel@he-arc.ch

Jonathan Dreyer
Haute École Arc Ingénierie HES-SO
St-Imier, Switzerland
jonathan.dreyer@he-arc.ch

Farid Abdalla
Haute École Arc Ingénierie HES-SO
St-Imier, Switzerland
farid.abdalla@he-arc.ch

Vicente Rodríguez Montequín
Project Engineering Area
University of Oviedo
Oviedo, Spain
montequi@uniovi.es

Zoltán Balogh
Department of Parallel and Distributed Information Processing
Institute of Informatics, Slovak Academy of Sciences
Bratislava, Slovakia
balogh@savba.sk

Emil Gatial
Department of Parallel and Distributed Information Processing
Institute of Informatics, Slovak Academy of Sciences
Bratislava, Slovakia
emil.gatial@savba.sk

Ivana Bundinská
Department of Parallel and Distributed Information Processing
Institute of Informatics, Slovak Academy of Sciences
Bratislava, Slovakia
ivana.bundinska@savba.sk

Adrian Gligor
Electrical Engineering and Information Technology Department
"George Emil Palade" University of Medicine, Pharmacy, Science and Technology of Targu Mures
Targu Mures, Romania
ORCID: 0000-0002-8724-4272

Laszlo Barna Iantovics
Electrical Engineering and Information Technology Department
"George Emil Palade" University of Medicine, Pharmacy, Science and Technology of Targu Mures
Targu Mures, Romania
barna.iantovics@umfst.ro, 0000-0001-6254-9291

Stefano Carrino
Haute École Arc Ingénierie HES-SO
St. Imier, Switzerland
stefano.carrino@he-arc.ch

***Abstract*—The aim of the Social Network of machines (SOON) project is to investigate the impact of using of autonomous social agents to optimize manufacturing processes in the framework of Industry 4.0. In this article, we present the multi-agent SOON architecture and the built solutions aiming at optimizing the scheduling of tasks. Two different scheduling approaches are proposed. The first approach is based on an 'auction' paradigm where the task assignment is decided according to the capability of a machine agent to bid for a task. The second approach is built on a heterarchical agents network where agents learn the acquisition of cooperative tasks. Both solutions are capable of managing and synchronizing the communication between agents while performing their tasks. To describe each approach, two industrial use cases are illustrated: wire rod mill manufacturing and mechanical part manufacturing. Finally, in the heterarchical network, agents are trained with reinforcement learning to maximize the cumulative reward and optimize the manufacturing scheduling. Results show that reinforcement learning allows learning the optimal behavior in multiple scenarios.**

Keywords— smart manufacturing, scheduling optimization, multi-agent architecture, auction paradigm, deep reinforcement learning

I. INTRODUCTION

The SOON project investigates the impact of using of autonomous social agents to optimize manufacturing processes in the framework of Industry 4.0. "Social" means that cyber-physical entities will act autonomously in order to optimize an industrial process following behavior models inspired by human social networks. Currently, in Industry 4.0, smart entities do exist [1]. However, intelligence is localized and intelligent heterogeneous entities cannot communicate together even inside the same shop-floor. Our motivation comes from the observation that, if we want to create a real Internet of Everything that brings together processes, data, things, and people, all these entities have to be connected and follow a shared, easy to understand paradigm.

The focus of the project SOON is to investigate how to deploy smart services on a social network of machines in order to deal with big data and optimize processes. In addition, SOON adds a crucial actor in this social network: The Human. In this sense, our work is closer to the concepts developed in [2] in which socio-technical networks composed of humans and things are investigated.

In this article, we propose a solution to optimize the network of autonomous smart machines seen as agents collaborating toward targeted tasks. We deal with the problem using two different scheduling approaches. First, a rule-based multi-agent approach where an 'auction' paradigm is applied. In this case, the task assignment is decided according to the capability of a machine agent to bid for a task. A centralized Poll Management and Aggregation System broker is hence required. Second, a heterarchical agents network where agents learn to cooperate

while performing their tasks. Deep reinforcement learning (RL) is applied so that agents learn in a supervised manner from previous attempts to take actions to maximize the cumulative return [3]. Both approaches are capable of managing and synchronizing the communication between agents while performing their tasks.

This paper is organized as follows, Section II presents an overview of the SOON CHIST-ERA project and the related work in the domain of multi-agent architectures. Section III introduces the Auction paradigms illustrated with the use case of the wire rod mill manufacturing. Section IV describes the RL environment in the use case of the mechanical part manufacturing. Results achieved with the RL solution are presented in Section V. Section VI concludes the paper highlighting the future work.

II. Related Work

A. SOON - Social Network of Machines

Global supply chains, market fragmentation, mass customization and shorter product life cycles have scaled up competition among companies, which give rise to the need for introducing cognitive abilities through flexible and easily reconfigurable production systems. In this sense, there is unmined potential for the European manufacturing industry to be more innovative, productive and competitive whilst using fewer resources and reducing environmental impact.

Industry 4.0 (I4.0) aims at addressing these points via the creation of network-centric production, new manufacturing techniques and Cyber-Physical Systems [4]. In modern Industrial Internet of Things (IIoT) solutions, data collection and processing can occur in the cloud or at device-side. Device-side processing is referred to as Edge (or Fog) computing [5]. Such technology could enable single entities, such as industrial machines to become autonomous, contextual-aware agents and lay the foundations for a multi-agent system. In such systems, agents are designed to decide to execute or not a requested task by considering its goals, priorities and the situation [4].

Atzori et al. [6] introduced the concept of Social Internet of Things based on a "social network of intelligent objects" in which the things in the network are connected via a "social" relationship. For instance, these relationships can be modelled following relations typical of social networks (friend of, following, etc.). In recent years (e.g., [7]), the researchers focused on the technological challenges such as network creation, scalability and trustiness. The focus of SOON project is to investigate how to deploy smart services on a social network of machines to deal with big data and optimize processes. In addition, SOON will add a crucial actor in this social network: The Human. In this sense, our work is closer to the concepts developed in [2] in which socio-technical networks composed of humans and "things" are investigated.

Adding the human component to a system that includes autonomous software agents is often considered necessary as well as desirable [8]. This paradigm is also called Human-Agent Collective (HAC) [9]. HACs are novel socio-technical systems in which humans and intelligent agents engage in flexible relationships to reach both their individual and collective goals. Humans and machines relationships can vary dynamically according to the task to be achieved and the environmental conditions: sometimes humans take the lead, sometimes the computer [10].

Although multi-agent software solutions have existed since several years, real applications in the industry are very limited and are often not tested or deployed [11]. Most of the algorithms have been developed in lab environments not dealing with the complexity of concrete applications and often are not tested in real environments with noise, disturbance, changing conditions, etc. [12]. We believe that autonomous decision taking systems should encompass human operators and share the work spaces. Humans have to be part of the equation. SOON has the potential to overcome these limitations defining a unified framework including machines and humans. Real world limitations will be addressed via the strict collaboration with companies moving into the I4.0 revolution.

B. Multi-agent approaches

Well-known reference architectures for I4.0 such as Reference Architecture Model Industry 4.0 (RAMI 4.0) [13], American Industrial Internet Reference Architecture (IIRA) [14] and 5C [15] offer a general framework and provide a conceptual starting point for the realization of IIoT-based multi-agent architectures.

Software agents take advantage of their social ability to exhibit flexible coordination behaviors that make them able to both cooperate in the achievement of shared goals or to compete on the acquisition of resources and tasks. Agents have the ability of coordinating their behaviors into coherent global actions. The first insights on the social-network properties came from Milgram's experiment [16], presenting a mail message routing task by communication with friends and local acquaintances (typically 5-6 in average). In the later work, the social-network of agents was researched mainly from the point of view of searching for the best acquaintances (e.g., in works [17] [18]).

The use-case for collaborating the agents is based on Real-time Bidding (RTB) [19] [20] where the agents (advertisers) bid for every individual impression in real-time when being generated. The goal is to find the best acquaintance among existing agents that is capable of efficiently performing the task in a given time. The agents usually keep the list of neighboring or acquaintance agents or may form agent communities which group the agents with similar features, i.e. location, domain type, task capabilities (Fig. 1).

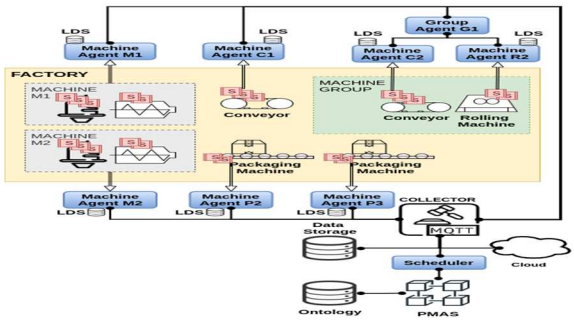

Fig. 1 Agent-based approach for Industry 4.0 use-case

C. Reinforcement learning

RL has become popular since the program AlphaGo defeated the human Go player champion Lee Sedol in 2016 [21] RL is a method of machine learning wherein the agent learns to perform certain actions in an environment which lead it to maximum reward. It does so by exploration and exploitation of knowledge it learns by repeated trials of maximizing the reward. Current methods to implement RL algorithms are value-based where long-expected returns of states are maximized; policy-based where a policy is created, such that the action performed at each state is optimal to gain maximum reward in the future; and model-based where the agent learns to perform in a specific virtual model environment [22]. When deep learning solutions (in contrast to 'shallow' learning) are combined with RL algorithms, the resulting approaches are often called "Deep reinforcement learning" [23]. Among RL algorithms, proximal policy optimization (PPO) is one of the most successful deep RL methods, achieving state-of-the-art performance across a wide range of challenging tasks.

In RL, complex environments can require agents to explore multiple configurations before they can begin to optimally exploit the environment and maximize rewards. Due to sparse reward signals, poor state representation, or the presence of adversaries, random exploration will not be able to reliably reinforce the actions taken to receive a reward [23]. Curriculum learning solutions suggest that learning can be accelerated by first training on a simpler task, and transferring the knowledge acquired to improve learning on a subsequent target task [3]. Curriculum learning considers how exactly to select and order different source tasks, such that performance or learning speed is improved on the target task.

III. AUCTION PARADIGM: USE CASE OF WIRE ROD MILL

Poll Management and Aggregation System (PMAS) is used to place a new order into production and employs ontology description to identify the agents which are capable of putting production tasks into the agent's internal schedule.

At the beginning, the production parameters of the new order are put as values to the poll template using PMAS to create a poll instance. The poll instance is passed to the Auction Broker which initializes the auction process. Usually several agents are registered to the auction which match their capabilities for placing the order. At runtime the Auction Broker may ask PMAS for an actual list of polls and initialize these polls as auctions. The Auction Broker schedules the auctions with a deadline and notifies registered agents that the auction was started. In the case a new agent was included to the system at runtime, it may ask for an auction list. During the auction process, the agents calculate and publish the bidding value to the Auction Broker which updates the auction status accordingly.

Auction Broker may update the auction status received from the PMAS (e.g., due to human interventions). When the auction reaches the deadline, the auction is closed by a finishing message and the results of the auction are sent to PMAS and to participating agents (Fig. 2). The winning agent will take over the auctioned task and put it into its own schedule. The auctioning system prototype was designed and developed with

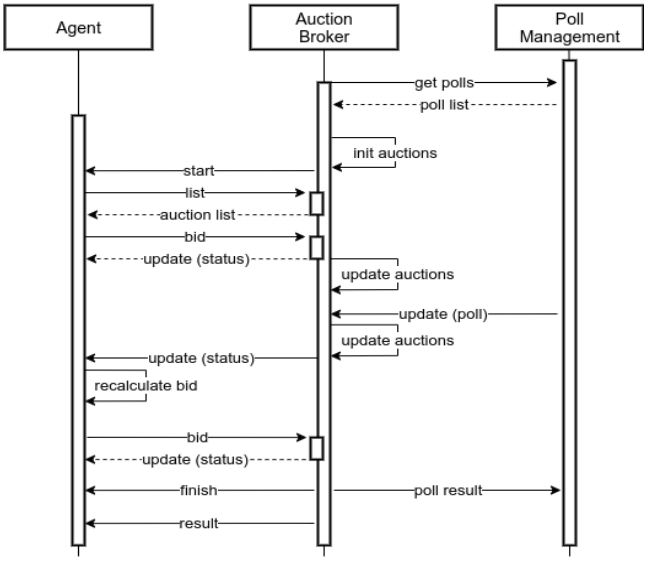

Fig. 2. Auctioning protocol sequence diagram

scalability and robustness in mind. The auction broker is implemented in Erlang and is using an EMQX broker. Erlang [erlang.org] is a programming language specifically built for massively scalable real-time systems. EMQX [emqx.io] is a scalable and reliable real-time MQTT message broker for IoT in the 5G Era.

Overall the main KPI is the "orders/batches completion time" and "failed orders reduction". The machines need to distribute the work in an optimal way to fulfill orders in required times.

Nevertheless each agent tries to optimize its own internal schedule taking into account possible outages. Therefore an agent's internal KPI can be different such as "maximizing own machine production resource utilization".

One of the scenarios for the evaluation of the auction paradigm is the rolling process in steel manufacturing. Particularly, a system for assisting technicians in the selection of the rolls for a wire rod mill is being developed. During the wire rod mill operation, the raw material goes through rolls to achieve the required cross-section profile. In the wire rod mill, the determination of the set of rolls for a specific batch of orders is a task of great importance. This is particularly complex as the number of assembled rolls in this kind of mill is very high, more than one hundred, and each of the single sections of the rolling mill has its specific requirements which depend on the kind of product to be rolled. Rolls have different tracks to be used and must meet diameter restrictions among rolls assembled in the same and successive stands.

The replacement of rolls is a regular task because when the rolls are worn, they must be disassembled and reconditioned so that they can be used again. This process reduces its diameter, and therefore, its lifetime. The wear the rolls suffer depends on several factors being the alloy of the roll, the tons of raw material rolled and the stiffness of the rolled material the most important. Each roll is an expensive asset because they are made with metal alloys prepared to work in hard conditions. Therefore, each roll life cycle is carefully managed until it is

finally disposed. The reduction of its diameter is measured and registered as it is one of the factors included for the computation of the production cost.

Nowadays, the technicians perform the operation replacing the assembled rolls with predefined sets of alike rolls working together. This simplifies the problem; however, it is not an optimal approach and makes the rolls replacing process less flexible. A Multi-agent System (MAS) using an auction-based negotiation approach has been designed to tackle this problem. The goal is to find the most suitable combination of rolls that could be used for a replacement accomplishing with the basic requirements and that entails the minimum cost. The cost is computed as the depreciation of the roll after performing its work, and is calculated considering the purchase value, the residual value, the initial diameter and the end-of-life diameter.

The auction negotiation approach is particularly useful for allocating scarce resources that are typically desired by more than one agent. In this scenario, the resources are the rolls. An auction mechanism is made up by a group of agents, where one agent has the role of auctioneer and the remaining agents are bidders. Each bidder agent codifies initially a stochastic solution for the replacement of rolls (a set of rolls that could be assembled in the rolling mill to perform the required job). The initialization process allocates rolls so that one roll can be included only in one solution. The solutions accomplish with the process and product constraints, so they are valid solutions but not efficient. Sets of rolls proposed by the technicians can also be included as other bidder agents. Hence, the algorithm considers *n* bidder agents that codify *n* possible feasible solutions.

The algorithm runs an iterative process. The auctioneer agent auctions the leftover rolls (those ready to be used for the job to be performed but not assigned to any of the bidder agents). Each bidder agent checks if the roll being auctioned can replace some of those included in their internal codification of the solution and, if so, the cost of performing the work. The cost is calculated using a predictive model for forecasting the wear of the rolls as described in [24]. If the cost is reduced, the bidder agent bids for the roll that amount. Only one bid can be done for each agent. The highest bid is the winner. Because each bidder agent contains a complete and valid solution, the new roll will replace one that will be returned to the pool of available rolls and that will be auctioned later. The process is repeated for all the rolls in the pool while there are rolls that have not been auctioned. When empty, a round is completed. The process runs again a new round. If no progress is made (no cost reduction is gained), the process finishes and the agent which codifies the solution with less cost is the winner. The process is depicted in Fig. 3.

IV. A HETERARCHICAL ORGANIZATION OF AGENTS: A REINFORCEMENT LEARNING SOLUTION

This section describes a multi-agent architecture for the optimization of task scheduling compatible with state of the art multi-agent RL algorithms. Differently from the auction paradigm, in this solution the agents should be able to learn how to optimize a manufacturing process without an established protocol and with a heterarchical organization. In order to be compatible with an RL solution, a manufacturing process has been modeled as a discrete Markovian Decision Process, in which a *step* represents the smallest discrete time unit. At each time step, every unit will take decisions autonomously knowing the status of the other machines and the order to be completed to maximize the cumulative reward.

A. Use Case: Manufacturing workshop

The scenario for the evaluation of the RL paradigm is a generic manufacturing plant in which multiple machines (agents) work together to fulfill one or multiple orders. In the proposed models, machine agents are organized in a workshop. A workshop is a place which provides both the area and tools required for the manufacturing of goods. The scenario is inspired from a fleet of machine tools.

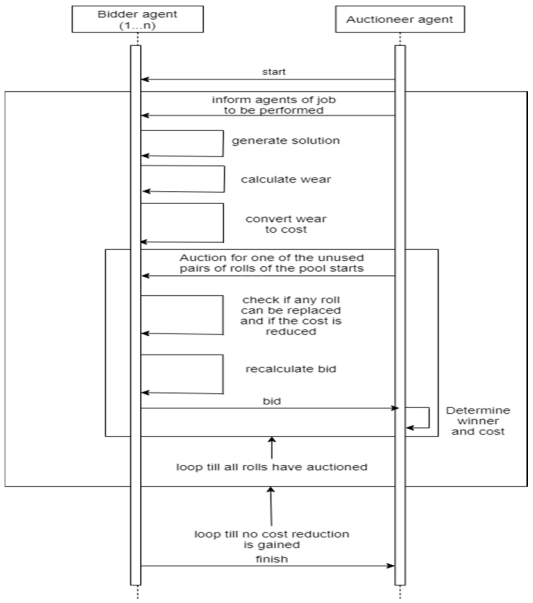

Fig. 3. Rolls replacement auctioning protocol sequence diagram.

The solution presented in this paper aims at finding the best succession of operations for each machine given a workshop layout and the order(s) to be completed.

B. Workshop simulator

RL algorithms need a large number of trial and errors before converging toward the optimal solutions. This means that a simulator is required to train the RL agents before their deployment into real machines. The proposed solution models a workshop as a combination of three components: machines, storages and links. A *machine* is characterized by the type of parts that can be processed and produced, the processing time to manufacture a part and the number of parts that can be worked in parallel (batch size). To switch from the production of a type of part to another a reconfiguration time is required (it represents the average time needed by an operator to reconfigure the machine to produce a different part type). In addition to machines, the workshop has a shared *storage* used to stock raw and processed materials. The last components are the *links* characterizing the time needed to move a batch of parts from/to a machine or the storage.

Finally, a workshop should be able to produce parts in response to input *orders*. Orders are characterized by a number of parts of different types to be produced before a time limit. The agents have knowledge of how to produce each part: for instance, agents know that to produce a part of type B_4 they need one part of type A_2 and one part of type A_3; to produce A_2 and A_3, raw material is required.

Therefore, a workshop is a succession of many different machines that process (produce, work, clean, etc.) parts to fulfill one or multiple orders. For instance, to have a finished part, we need to create a piece (A), to heat it (B), then clean it (C). For simplification and generalization, this manuscript will only refer to the sequence of production, e.g., $A_1 \rightarrow B_1 \rightarrow C_1$ and not the detail of what the machine does. Fig. 4 illustrates a simple workshop layout with seven different machines linked to create final parts of type C1 and C2.

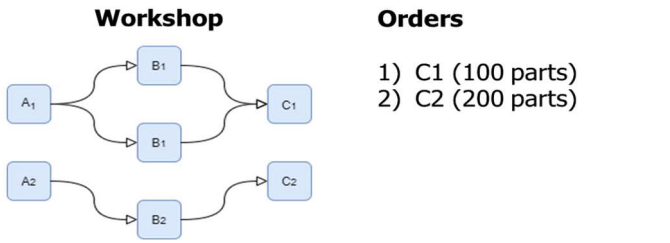

Fig. 4. Example of a workshop layout.

Fig. 5. Machine reconfiguration.

The following scenarios are based on the hypothesis that a machine can elaborate and produce different types of parts. However, a reconfiguration step is needed to model the time required to reconfigure a machine to process another type of parts. For instance, a machine that creates screws (A_1), needs to be reconfigured before creating nails (A_2) - as depicted in Fig. 5 (in regard to time, this is typically a very costly operation). Before reconfiguration, the first machine produces parts of type A1 and the second machine of type B1; after reconfiguration, the first machine starts producing A2 parts and the following machine B2 parts.

To summarize, at each time step a machine has to autonomously decide which action to perform among a discrete set of actions: "do nothing", "reconfigure to type X", "get pieces from the storage", and "produce a new part".

C. Implementation

Every machine in the workshop is implemented as an RL agent with different, configurable actions and observations.The agent has been implemented using PPO (Proximal policy optimization) algorithm [22]. Multiple training strategies have been evaluated. In particular, curriculum learning strategy has been developed in order to improve the training of the AI agents in progressively harder problems.

V. RESULTS

In order to evaluate the proposed architecture, a simple workshop configuration has been realized. Fig. 6 illustrates the process flow of the evaluated workshop configuration. The number on the arrows describes the number of time steps required to transfer a part from the storage to a machine and from a machine to another machine (1). The number inscribed in the circle details the time required to produce a part (e.g., 1 time steps for a part of type A1, 10 time-steps for a part of time B1). Reconfiguration time, not depicted in the figure, is of 5 time steps for each machine. A machine of type A can produce parts of type A_1, A_2 or A_3; A_1 and A_2 parts are needed by a second machine (B) to produce parts of type B_1 and B_2. In this scenario, producing A_3 or B_2 parts would be a waste of resources (time and raw material) since they are not demanded by the orders. In fact, the objective for this scenario is to produce 50 parts of type B_1 in an optimal manner.

A 'vanilla' solution in which the RL agents have to learn the final optimal policy has been compared with a curriculum learning solution. The curriculum learning solution consists in progressively increasing the complexity of the environment in order to facilitate the agents' learning. In our scenario we start with an environment where the storage already contains enough A_1 parts (50) to produce and fulfill the B_1 order. Then, after a fixed number of time steps, we change the initial workshop with a slightly more complicated configuration, where only 40 A_1 parts are present. The next configuration only holds 20 A_1 parts

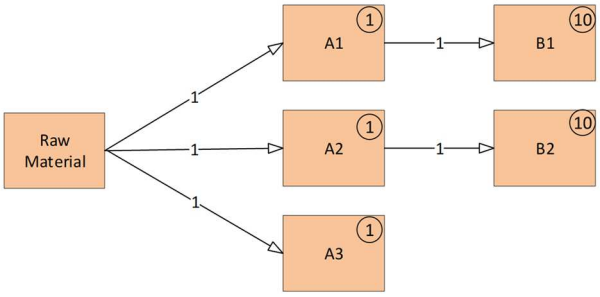

Fig. 6. The evaluated workshop configuration

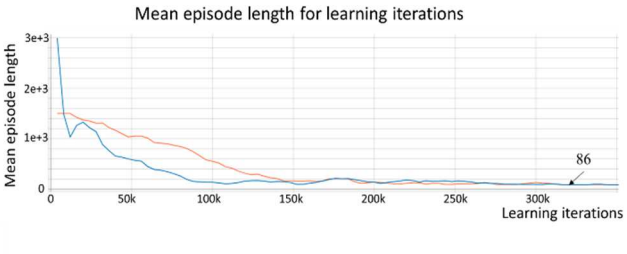

Fig 7. Average episode length with no configuration required.

In the storage, and the final configuration starts without any parts except for the raw material. That means that the second agent (producing B_1 parts) needs to totally rely on the first machine to get the required parts. Moreover, compared to the

last configuration, the initial workshop doubles the maximum number of time steps before resetting the environment allowing the agents to try more configurations during each episode.

Fig. 7 shows the mean number of steps required to complete (or fail) the orders. In orange is the vanilla solution, in blue the solution using curriculum learning. While both solutions converge to the global optimum, the curriculum learning solution is, on average, 1.5 times faster to converge.

VI. CONCLUSION

In the design of the SOON architecture, we have faced diverse challenges. One of them consists in the fact that the integration of heterogeneous subsystems is difficult to handle, this emerged from the very different profile of the industrial partners. Based on the complexity of the industrial plants, an initial development is gradually transformed into a fully functional smart cyber-physical production system. Over time, additional services can be added to the system. Included services can span multiple layers of the initial system. This extension must be made in such a way as to not break the compatibility with the existing system.

The auctioning system presented in the paper is demonstrated in the factory with several parallel production lines that may take over the production batch from each other. Therefore the auction-based approach described in the paper is suitable mainly for settings where there are several similar machines capable of competing for batches or orders. However the auctioning can be set up to suit other manufacturing challenging setups. Auction system may provide the history of the past auctions so that the agents may check what the best possible solution was in the past - such data can be then indeed used to train the RL agent.

In future, we are planning to test the proposed architecture with more complex and dynamic scenarios. As illustrated in Fig. 8, machine failures will be included in the simulator. (Left) The workshop is optimized and produces parts C1 and C2. (Center) The machine producing parts B2 fails and stops the production. (Right) A similar machine automatically decides to reconfigure itself to replace the missing machine.

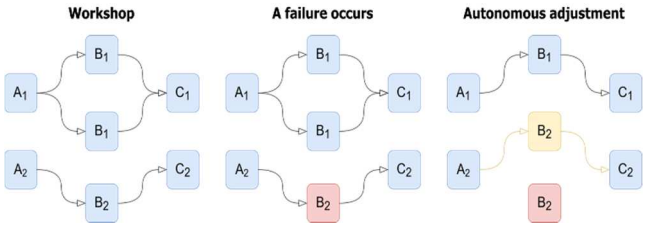

Fig. 8. Workshop reconfiguration in the case of failure

ACKNOWLEDGMENT

This work was supported by the CHIST-ERA grant CHIST-ERA-17-BDSI-006, by Swiss National Fund (SNF), project number 20CH21_180431, by Slovak Research and Development Agency under the contract No. 2/0125/20, by the Romanian National Authority for Scientific Research and Innovation, CCCDI-UEFISCDI, 101/2019, COFUND-CHISTERA-SOON, within PNCDI III, and by AGENCIA ESTATAL DE INVESTIGACION (Spain), grant number MCIU-19-PCI2019-103443.

REFERENCES

[1] SCOOP. Industry 4.0: the fourth industrial revolution – guide to Industrie 4.0. (https://www.i-scoop.eu/industry-4-0/).

[2] Kranz, M., Roalter, L. and Michahelles, F., 2010. Things that twitter: social networks and the internet of things. In What can the Internet of Things do for the Citizen (CIoT) Workshop at The Eighth International Conference on Pervasive Computing (Perv. 2010) (pp. 1-10).

[3] Narvekar, S., Peng, B., Leonetti, M., Sinapov, J., Taylor,M. E., and Stone, P. 2020. Curriculum learning for reinforce-ment learning domains: A framework and survey.arXivpreprint arXiv:2003.04960

[4] Erol, R., Sahin, C., Baykasoglu, A. and Kaplanoglu, V., 2012. A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems. Applied soft computing, 12(6), pp.1720-1732.

[5] Shi, W., Cao, J., Zhang, Q., Li, Y. and Xu, L., 2016. Edge computing: Vision and challenges. IEEE Internet of Things Journal, 3(5), pp.637-646.

[6] Atzori, L., Iera, A. and Morabito, G., 2011. Siot: Giving a social structure to the internet of things. IEEE communications letters, 15(11), pp.1193-1195.

[7] Nitti, M., Girau, R. and Atzori, L., 2014. Trustworthiness management in the social internet of things. IEEE Transactions on knowledge and data engineering, 26(5), pp.1253-1266.

[8] Kamar, E. 2016. Hybrid workplaces of the future. XRDS: Crossroads, The ACM Magazine for Students 23(2):22–25.

[9] Jennings, N.R., Moreau, L., Nicholson, D., Ramchurn, S., Roberts, S., Rodden, T. and Rogers, A., 2014. Human-agent collectives. Communications of the ACM, 57(12), pp.80-88.

[10] Pticek, M., Podobnik, V. and Jezic, G., 2016. Beyond the Internet of Things: The Social Networking of Machines. International Journal of Distributed Sensor Networks, 12(6).

[11] Wang, S., Wan, J., Zhang, D., Li, D. and Zhang, C., 2016. Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination. Computer Networks, 101, pp.158-168.

[12] Leitão, P., 2009. Agent-based distributed manufacturing control: A state-of-the-art survey. Engineering Applications of Artificial Intelligence, 22(7), pp.979-991.

[13] Deutsches Institut für Normung eV (2016). Reference architecture model industrie 4.0 (RAMI4.0) (DIN SPEC 91345)

[14] Industrial Internet Consortium (2017). The industrial internet of things. Vol. G1: reference architecture. https: //www.iiconsortium.org/IIC_PUB_G1_V1.80_2017-01-31.pdf

[15] Lee, J., Bagheri, B., Kao, H.A. (2015). A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manuf. Lett. 3:18-23.

[16] Milgram, S.,The small world problem. Psychology today, 1 (1), pp. 61-67, 1967.

[17] Yu, B., & Singh, M., Searching social networks. Proceedings of the second international joint conference on Autonomous agents and multiagent systems, pp. 65-72, 2003.

[18] Franchi, E., A Multi-Agent Implementation of Social Networks. Undicesimo Workshop Nazionale "Dagli Oggetti agli Agenti" (WOA 2010), pp. 1-6, Rimini, 2010.

[19] Jun Wang, Weinan Zhang, and Shuai Yuan. 2016. Display advertising with real-time bidding (RTB) and behavioural targeting.arXiv:1610.03013(2016).

[20] Yuan, Yong & Li, Juanjuan & Qin, Rui. (2014). A survey on real time bidding advertising. Proceedings of 2014 IEEE International Conference on Service Operations and Logistics, and Informatics, SOLI 2014. 418-423. 10.1109/SOLI.2014.6960761.

[21] D. Silver et al., "Mastering the game of Go without human knowledge", Nature, vol. 550, no 7676, p. 354-359, oct. 2017, doi: 10.1038/nature24270.

[22] Sutton, R. S., & Barto, A. G. 2018. Reinforcement learning: An introduction. MIT press.

[23] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. 2018. Deep reinforcement learning that matters. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

[24] Junquera, A. M. V., González, J. G., Balsera, J. M. V., & Montequín, V. R. (2020). A Wire Rod Rolling Mill Digital Twin for the Simulation of the Rolls Replacement Process. In Multidisciplinary Digital Publishing Institute Proceedings (Vol. 63, No. 1, p. 13).