Epureanu, B. I., Li, X., Nassehi, A., & Koren, Y. (2020). Self-repair of smart manufacturing systems by deep reinforcement learning. *CIRP Annals - Manufacturing Technology*.
<https://doi.org/10.1016/j.cirp.2020.04.008>

Peer reviewed version

License (if available):
CC BY-NC-ND

Link to published version (if available):
[10.1016/j.cirp.2020.04.008](https://doi.org/10.1016/j.cirp.2020.04.008)

[Link to publication record in Explore Bristol Research](#)
PDF-document

This is the author accepted manuscript (AAM). The final published version (version of record) is available online via Elsevier at <https://www.sciencedirect.com/science/article/abs/pii/S0007850620300299> . Please refer to any applicable terms of use of the publisher.

University of Bristol - Explore Bristol Research

General rights

This document is made available in accordance with publisher policies. Please cite only the published version using the reference above. Full terms of use are available:
<http://www.bristol.ac.uk/red/research-policy/pure/user-guides/ebr-terms/>

CIRP Annals - Manufacturing Technology

Self-repair of smart manufacturing systems by deep reinforcement learning

--Manuscript Draft--

Manuscript Number:	2020-O-40R2
Article Type:	STC O
Keywords:	Reconfiguration, Decision making, Artificial intelligence
Corresponding Author:	Xingyu Li, Ph.D University of Michigan Ann Arbor, Michigan UNITED STATES
First Author:	Bogdan Epureanu, Professor
Order of Authors:	Bogdan Epureanu, Professor
	Xingyu Li, Ph.D
	Aydin Nassehi, Professor
	Yoram Koren, Professor
Abstract:	<p>Smart manufacturing is built on complex decision-making in real-time based on data from networked machines and sensors. As a potential enabler of smart manufacturing, reconfigurability enhances adaptability to demands and enriches the utility of the collected data. This study focuses on a synergistic combination of advanced manufacturing technologies in a highly diverse market environment, to identify deficient components by inferring from changes in product quality and to sustain the operation of the manufacturing system by creating multiple-level self-repair strategies. Deep reinforcement learning is then used to select the strategy based on system status and performance.</p>

Contents lists available at [ScienceDirect](#)

CIRP Annals Manufacturing Technology

Journal homepage: www.elsevier.com/locate/cirp

Self-repair of smart manufacturing systems by deep reinforcement learning

Bogdan I Epureanu^a, Xingyu Li^{a,*}, Aydin Nassehi (2)^b, Yoram Koren (1)^a^aDepartment of Mechanical Engineering, University of Michigan, Ann Arbor, USA^bDepartment of Mechanical Engineering, University of Bristol, UK

*Corresponding author

Smart manufacturing is built on complex decision-making in real-time based on data from networked machines and sensors. As a potential enabler of smart manufacturing, reconfigurability enhances adaptability to demands and enriches the utility of the collected data. This study focuses on a synergistic combination of advanced manufacturing technologies in a highly diverse market environment, to identify deficient components by inferring from changes in product quality and to sustain the operation of the manufacturing system by creating multiple-level self-repair strategies. Deep reinforcement learning is then used to select the strategy based on system status and performance.

Reconfiguration, Decision Making, Artificial Intelligence

1. Introduction

Smart manufacturing systems (SMSs) are advanced intelligence systems which promote responsiveness to changes in the market environment and changes in system condition by adopting networked data, information and communication technologies [1]. Recent breakthroughs in cyber-physical systems, artificial intelligence (AI), industrial internet of things (IIoT), and cloud computation enable deployment of SMSs [2] and make the vision of fully automated manufacturing systems possible [3]. The performance of SMSs is characterized by behaviors, including self-organization, self-diagnosis, and self-healing [4] traditionally associated with biological systems. Due the rapid pace of technological development and product personalization, future SMSs have to be highly adaptable. Adaptability can be realized by machines that can be decomposed into swappable components, i.e. modules [5], and reconfigured to respond to mass-individualized demands [6, 7]. Such flexibility of configuration [8] has been proven to raise responsiveness to demands, to promote interoperability [9,10], and to enhance the sustainability [11, 12].

In particular, trouble-free operation of components is critical for system performance [12, 13] in modern large-scale automated manufacturing systems. Difficulties in locating faults create production delays, system downtime and increased total production cost. Thus, system diagnostic and remedy techniques are needed for modern manufacturing systems. Classic diagnosis methods include fault detection and isolation, and failure mode and effects analysis. These methods match system symptoms to component failures, and thus are hard to implement in complex systems with many failure modes [18]. Probabilistic and learning-based approaches can be used to calculate failure probabilities based on the production history, such as Bayesian networks [20] and neural networks [19]. Manufacturing personalized products require companies to have a continuous refinement of their manufacturing processes and system configuration. The

production environment requires diagnosis models to be adaptive to system changes and unexpected demands. The large variety of products also limits historical operation data available per product. Thus, the performance of existing models is limited. Methods to combine real-time fault diagnosis and repair strategy selection in highly-dynamic production environments remains to be addressed.

Approaches in Industry 4.0 promise to address this challenge by increased use of IIoT technologies and diverse sensors to capture data at all stages of a product's life. Flexibility in product, operation, and configuration makes the quality stream customizable in a way that enables testing and verifying the condition of suspect components. With proper usage of quality data, SMSs are capable of self-diagnosis and self-repair to make the overall system operation sustainable under system faults.

An important challenge emerging is to create methods to use product/parts diagnosis information to perform system fault diagnosis, including: 1) the connection between product diagnosis and system faults is unclear and varies in different systems and configurations; 2) the system is uncertain, i.e., time and the cost for reconfiguring the manufacturing system depend on the operator expertise and technology; 3) the complexity of the decision process creates latency when centralized decision-making algorithms are used. Thus, pushing computation from the network core to the production line becomes a necessity [18]. That requires a local decision-making algorithm that model and optimize the repair policy according to the limited information.

Herein, a system self-diagnosis and self-repair approach are designed to enable a fully-automated factory to sustainably operate despite system faults. Three strategies are proposed for inferring the deficient components of the overall system through real-time optimization and graph theory. An AI decision-maker enabled by a deep convolutional reinforcement learning network is created to dynamically select a strategy to deal with deficient components according to the configuration information, and to evolve the repair policy learning from past system performance.

2. Self-Repair Strategies

Product personalization poses new challenges in the management of SMSs and brings new opportunities in system diagnosis. The core idea is to intelligently use different products that require diverse operations and, when possible, reconfiguration of the manufacturing system to link quality deterioration to system faults. As an example, consider that demands are for diverse types of parts to be processed by a manufacturing system, which comprises multiple stages, each with a number of machines made of modules. Each part contains multiple features. Each feature requires operations from at least one machine. All feature qualities are assumed to be monitored in the finished products. If a feature is processed by a machine with a deficient component, a deterioration of quality occurs. The deficient modules need to be removed once identified, and deficient machines can only be used after module repair.

Once a deficient feature is detected, system interrogation operation starts by changing the operations and products to gather information and determine the deficient stage(s). Focusing first on stages instead of modules significantly shrinks the search space. To maximize effectiveness in stage detection, the suspected stages are required to process a unique feature. The stages which process features that no other stage processes are defined as unique-stages. An example of part and operation rescheduling is shown in Fig. 1.

Figure 1. An example of deficient stage detection through the autonomous part change and operation rescheduling.

The capacity of the system to be tested is important because a deficient feature can only be verified after a certain amount of production is reached. The capacity $spr_{k,n}$ of stage n is determined by the operation $op_{k,n}$ and stage index (with corresponding machine types). The minimum stage capacity defines the line capacity pr_k . Given part type as k and unique stages set as $\mathbf{u}$, a model is formulated to assign the interrogation operations that maximizes the line capacity, namely

$$\begin{aligned} \min_{op_k} & -pr_k \\ \text{s.t. } (a) & \mathbf{mp}_k^T \mathbf{op}_{k,n} \leq \sum_{\tilde{n}=1}^{\tilde{n}=n-1} \mathbf{mc}_k^T \mathbf{op}_{k,\tilde{n}}, \quad \forall n, \forall k \\ (b) & pr_k = \min_{\tilde{n}} spr_{k,\tilde{n}}, \quad \forall n, \forall k \\ (c) & \mathbf{mf}_k^T \mathbf{op}_{k,n} \neq \mathbf{mf}_k^T \mathbf{op}_{k,\tilde{n}}, \quad \forall n, \forall \tilde{n} \in \mathbf{u} \\ (d) & , \quad \forall n, \forall k, \end{aligned} \quad (1)$$

where $\mathbf{mp}_k$, $\mathbf{mc}_k$, $\mathbf{mr}_k$ are precedence, element, and competence matrix defined in [9]. $\mathbf{mf}_k$ is the mapping between features and operations. Constraints (a) and (b) ensure the operations are completed following the precedence constraint; (c) ensures that the capacity of the line is the minimum capacity of stages; (d) ensures the features processed by unique-stages are different from the other stages; (e) specifies the bounds and type of decision variables.

The model is a mix-integer programming solved using CPLEX. By denoting the capacity $pr_k = f_{pr}(k, \mathbf{op}_k, \mathbf{u})$, product type k_t and operations $\mathbf{op}_{k_t}$ used in the interrogation operation is

$$k_t, \mathbf{op}_{k_t} = \underset{k}{\operatorname{argmax}} f_{pr}(k, \mathbf{op}_k, \mathbf{u}). \quad (2)$$

Rescheduling stops when only one suspect stage remains. Given the identified deficient stage, the deficient machine can be located by manipulating the capacity of machines, i.e., reduce one by half, and matching the percent of deficient parts to the machine capacity. In conventional machine repair strategies, domain expertise is required for inspection, repair, and replacement, which may create significant system down-time. Thus, another two reconfiguration-based strategies are proposed assuming that modules are swappable in the SMSs. These strategies are referred as the speed-oriented module swap strategy and the capacity-oriented module swap strategy. To evaluate these strategies, capacity losses are selected as a measure of system performance.

To control the loss of capacity during diagnosis, the time to accomplish module swapping needs to be minimized. The suspected modules need to be assigned to unique-stages as much as possible, which leads to the speed-oriented module swap strategy. Moreover, after the removal of the deficient module, the capacity of the remaining manufacturing system also needs to be maximized. For example, the system will totally lose all capacities if the deficient component is located at a stage with a single machine (i.e., a single-machine stage). The repair of the component will make the unique machine unavailable. Thus, the overall configuration line loses the capacity totally until the module is returned. As an alternative, the capacity-oriented strategy is to swap the modules to the non-single-machine stages.

The maximum number of unique-stages in a configuration plays an important role in a swap decision. Because of the importance of the capacity, a swap should maximize the number of features that can be processed at unique-stages. As modules of the same type can be swapped, the swappable stages s_w are defined as the stages that can swap their modules with a deficient machine, the maximal unique-stages $\mathbf{u}$ is calculated by solving the following global optimization model using evolutionary algorithm

$$\begin{aligned} \max_{\mathbf{u}} & \text{len}((\mathbf{u} \cap s_w)) * \sum pr_k \\ \text{s.t. } & pr_k = f_{pr}(k, \mathbf{op}_k, \mathbf{u}). \end{aligned} \quad (3)$$

Before diagnostic, all modules of a suspect machine are suspect modules. Each suspect module d_i is desired to be swapped to a unique-stage u_j . The swap is rewarded by a value $c(d_i, u_j)$. For the speed-oriented swapping strategy, the reward is $c(d_i, u_j) = \frac{n_{m,u_j}}{t_{p,d_i,u_j}}$, where n_{m,u_j} is the number of machine at stage u_j , and t_{p,d_i,u_j} is the time required to swap suspect module d_i to stage u_j . For the capacity-oriented swapping strategy, the reward is $c(d_i, u_j) = \frac{n_m}{t_{swap}}$ if $n_m > 1$ and -1 otherwise. Suspect modules and available stages are nodes D and U in a graph $G(D, U, E)$ formulated by using edges E which link suspect modules and unique-stages. Reward $c(d_i, u_j)$ is assigned to edge $E(d_i, u_j)$. To maximize the weighted sum without duplicated stages, the graph is solved as a bipartite maximum matching problem. An example of the matching graph is shown in Fig. 2.

Figure 2. An example graph for suspect module to unique-stages matching.

Figure 3. Deep convolutional Q learning network for self-diagnosis of SMSs; inputs obtained from swapping provide information to define repair strategies.

3. Strategy Selection

Each repair strategy has pros and cons. For example, the long repair and inspection time in a machine repair strategy makes the system operate with missing machines for a long time. However, for stages with large numbers of machines, the capacity loss can be small. Alternately, swap strategies can shorten the inspection time by simple swaps of modules and products. Compared to the speed-oriented strategy, the capacity-oriented strategy ensures that deficient modules are located at non-unique-stages to avoid total capacity loss. However, this strategy increases the swap time, especially for configurations with very few swappable stages. The selection of the strategy is challenging not only because of the complex inference process that starts from deficient features, but also because of the large number of diverse modules and machines, which require different time for swapping and relocating. More importantly, not all the parameters are easily measured, e.g., inspection times change factory by factory. As a consequence, a self-learning decision-making method is critical for the SMSs.

Reinforcement learning is a promising solution, which can selectively explore an unknown environment and exploit the past experiences, i.e., tuples of action a , state s , and reward r , to optimize a policy that maximizes the expected value of rewards over successive steps [18]. In this study, an action a is one selection of one of the repair strategies. The states of the system include current suspect modules, swappable stages, module types, and number of modules at each stage. The reward is the negative of the amount of capacity loss during repair. Infeasible actions are penalized by a large capacity loss. In this study, the policy can be a pure repair strategy or a mixed strategy.

A deep convolutional Q network is used to analyze complex input features in order to learn the optimal policy from the system performance. A value Q is used to represent the quality of an action given the state, which is updated as

$$Q(x, a) \leftarrow (1 - \alpha)Q(x, a) + \alpha[r(x, a) + \gamma \max_{a'} Q(x', a')], \quad (4)$$

where x' is the next state given the selected strategy a' , α is the learning rate, γ is the discount factor of the past experience. The Q value table is replicable by a neural network M , i.e., $Q(x, a) = M(x, a, \theta)$. The model parameter θ is updated by cost function J ,

$$J = [Q(x, a) - (r(x, a) + \gamma \max_{a'} M(x', a', \theta))]^2. \quad (5)$$

Details of the networks designed for repair strategy selection are shown in Fig. 3. The convolutional layers help the AI approach to capture contextual information of the system status. The softmax function is used as the policy for training, where temperature T is used to adjust the exploration and exploitation; a high temperature makes the decision-making more random.

4. Prototype implementation

An example from previous studies [9,12] is used to demonstrate the implementation of the approach. A simulation

model is built to simulate the operation of an autonomous manufacturing line with seven stages to produce parts of different types. Two types of parts with 9 and 14 features are selected and required to be processed by a predefined sequence of operations. Quality information is available for all features for part diagnosis. Changes in the system, i.e., operation rescheduling and part changes, are tracked and used in the product quality re-analysis. The product diagnosis starts once the number of parts reaches a threshold, which is 50. Deficient features trigger system diagnosis.

Through system diagnosis, deficient machines/modules are identified by the selected repair strategy and moved to the repair center for recovery. The time for inspecting a module is 30 min and for repairing a module is 20 min. The assembly time is 10 min for machine tools and 60 min for bases or arms. Disassembly times are half of the assembly times. The time required for each repair strategy is the sum of the times of all required actions. To satisfy the demands, the system is required to continuously provide capacity of at least 180 parts/hour. Lost capacity due to the repair strategy is calculated once the repair is completed. The learning rate, temperature, and discount factor are 0.01, 1, and 0.99.

4. Results

The performance of the three strategies is compared in a selected configuration. The modules of different types (shown in different colors in Fig. 4) highlight the configuration composition and the commonality of modules among stages. Only modules in same type (color) are swappable. The lost capacities of three repair strategies are calculated by iteratively selecting one module as the deficient one, the averaged lost capacities by stage are estimated by a simulation model and compared between strategies in Fig. 5.

Figure 4. The configuration used for strategy comparison, including 18 spindle 1, 16 spindle 2, 10 base 1, 4 base 2, and 30 axes.

Figure 5. Comparison of lost capacities between different repair strategies

The machine repair strategy leads to less capacity losses for a stage with many machines, where other machines can reschedule to process a different feature to preserve capacity during machine inspection. The machine repair strategy is the only choice when no module swapping is possible, e.g., at stage 6. Compared to the machine repair strategy, module swapping strategies require less

time to repair stages with few machines and affect capacity in a stable manner, around 300 parts (~1.5-hour of production). This value is weakly impacted by the stage. Thus, swap strategies have a more predictable behavior. The speed-oriented module swap strategy performs slightly better than the capacity-oriented module swap strategy. The large number of single-machine stages, i.e., 3 of 7, significantly raises the number of swaps for the capacity-oriented swap strategy to locate deficient modules.

Next, the performance of the reinforcement learning are also tested by repeatedly damaging the module of index 20, which is initially in stage 4. Due to module swapping, the module of index 20 keeps changing its location, which requires the repair strategy to be changed accordingly. Changes in the averaged lost capacity are shown in Fig. 6. The faulty module location is shown in red.

Figure 6. Changes of capacity and location of the module in index 20

Each step in Fig. 6 represents a repair period, which starts from a defective feature and ends at a completed repair. After a period of trial-and-error learning, the deficient module of index 20 finally settles at stage 2 and the system keeps adopting strategy 1. According to Fig. 4, strategy 1 at stage 2 preserves all the system capacity. Thus, repeating strategy 1 is the best policy for stage 2. The AI identifies this best repair policy; moreover, it also places the problematic module at the low-impact location.

By simulating a repeated deficiency at a random module after a repair, the lost capacity is compared in Fig. 7 between the optimal policy from deep learning and a random strategy selection. In the first 250 steps, the AI learns from negative experiences, when it keeps selecting inefficient strategies. However, the lost capacity reduces quickly and converges to 310 parts at step 1400. The capacity loss in the random strategy selection is over 600 parts. The learned policy is shown in Fig. 8.

Figure 7. Comparison of the performance between the optimal policy from the deep convolutional Q learning and random selection policy.

Figure 8. Percentage of strategies selected at different stages.

The diversity in module types and locations make the optimal policy a mixed strategy. Without knowing the action time, plant layout, and system capacity, the AI frequently adopts a module swap strategy for single-machine stages, a machine repair strategy for stages with multiple machines, and an infeasible swap for stages with no swappable modules, i.e. stage 6. The AI has the ability to evolve policies in reacting to system changes, i.e., reconfiguration time, inspection time, and system capacity.

5. Conclusions

This paper investigates the potentials of self-repair ability of future smart manufacturing systems integrating several advanced approaches, namely reconfigurability, edge computing, data-driven analysis, and deep learning networks. The paper offers a potential solution for self-repair as enabler of sustainable operation of smart manufacturing systems in a fully automated factory under a mass-personalized and dynamic production and market environment. Autonomous self-repair strategies are proposed by utilizing product and module swapping, operation rescheduling, and system reconfiguration. An AI-based decision-making algorithm is proposed for adaptively selecting a repair strategy by using a deep convolutional Q learning neural network. Simulation results show that the designed AI successfully converges to an optimal policy, which significantly reduces the capacity loss compared to a random strategy selection. Promising future research directions include: methods to achieve self-diagnosis for time-varying demands and configurations; and additional strategies informed by inspection and reconfiguration costs.

References

- [1] Qu Y J, Ming X G, Liu Z W, et al., 2019, Smart manufacturing systems: state of the art and future trends, *International Journal of Advanced Manufacturing Technology*, 103: 3751-3768.
- [2] Mittal S, Khan M A, Romero D, et al., 2019, Smart manufacturing: characteristics, technologies and enabling factors, *Journal of Engineering Manufacture*, 233(5): 1342-1361.
- [3] Davis J, Edgar T, Graybill R, et al., 2015, Smart manufacturing, *Annual Review of Chemical and Biomolecular Engineering*, 6: 141-160.
- [4] Park H S, Tran N H, 2012, An autonomous manufacturing system based on swarm of cognitive agents, *Manufacturing Systems*, 31(3): 337-348.
- [5] Colledani M, Angius A, 2019, Integrated production & reconfiguration planning in modular plug-and-produce production systems, *CIRP Annals*, 68(1): 435-438.
- [6] Koren Y, Heisel U, Jovane F, Moriawaki T, Ulsoy G et al., 1999, Reconfigurable Manufacturing Systems, *CIRP Annals*, 48(2): 527-540.
- [7] Koren Y, Shpitalni M, Gu P, Hu S, 2015, Product Design for Mass-Individualization, *Procedia CIRP*, 36: 64-71.
- [8] Minguillon F E, Schomer J, Stricker N, Lanza G, Duffie N, 2019, Planning for changeability and flexibility using a frequency perspective, *CIRP Annals*, 68(1): 427-430.
- [9] Li, X., Bayrak, A. E., Epureanu, B. I., Koren, Y., 2018, Real-time teaming of multiple reconfigurable manufacturing systems, *CIRP Annals*, 67(1):437-440.
- [10] Ma A, Nassehi A, Snider C., 2019, Anarchic manufacturing, *International Journal of Production Research*, 57(8): 2514-2530.
- [11] Koren Y, Gu X, Badurdeen F, et al., 2018, Sustainable living factories for next generation manufacturing, *Procedia Manufacturing*, 21: 26-36.
- [12] Li X, Nassehi A, Epureanu B I, 2019, Degradation-aware decision making in reconfigurable manufacturing systems, *CIRP Annals*, 68(1):431-434.
- [13] Zhou Z D, Chen Y P, Fuh J Y H, et al., 2000, Integrated condition monitoring and fault diagnosis for modern manufacturing systems, *CIRP Annals*, 49(1): 387-390.
- [14] Nguyen D T, Duong Q B, Zamai E, et al. Fault diagnosis for the complex manufacturing system. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 2016, 230(2): 178-194.
- [15] Don, M. G., & Khan, F. (2019). Dynamic process fault detection and diagnosis based on a combined approach of hidden Markov and Bayesian network model. *Chemical Engineering Science*, 201, 82-96.
- [16] Suh J H, Kumara S R T, Mysore S P. Machinery fault diagnosis and prognosis: application of advanced signal processing techniques. *CIRP Annals*, 1999, 48(1): 317-320.
- [17] Kruger J, Lehr J, Schluter M, Bischoff N, Deep learning for part identification based on inherent features, *CIRP Annals*, 68(1): 9-12.
- [18] Lin C C, Deng D J, Chih Y L, et al., 2019, Smart manufacturing scheduling with edge computing using multi-class deep Q network, *IEEE Transactions on Industrial Informatics*, 15(7): 4276-4284.