Production Engineering (2019) 13:33–41
https://doi.org/10.1007/s11740-018-0855-7

PRODUCTION MANAGEMENT

Reinforcement learning for opportunistic maintenance optimization

Andreas Kuhnle[1] · **Johannes Jakubik**[1] · **Gisela Lanza**[1]

Received: 3 July 2018 / Accepted: 22 October 2018 / Published online: 29 October 2018

Abstract
Intelligent systems, that support the maintenance of production resources, offer real-time data-based approaches to optimize the maintenance effort and to reduce the usage of resources within production systems. However, unused potentials remain regarding maintenance schedules with minimal opportunity costs of the measures taken. This work provides a novel, machine-learning-based approach for the exploitation of these remaining optimization opportunities as an exemplary extension of the current state of the art. The determination of an optimal maintenance schedule for parallel working machines, is based on the data of a production system. The main result of this work is the performance of the implemented reinforcement learning algorithms, both in terms of downtime reduction, which increases the production output, and in terms of reducing maintenance costs compared to existing maintenance strategies. Hence, this work provides a holistic approach to the optimization of maintenance strategies and gives further evidence of a meaningful applicability of reinforcement learning algorithms in manufacturing processes.

Keywords Reinforcement learning · Opportunistic maintenance · Opportunity cost reduction · Multi-agent-systems · Proximal policy optimization · Production planning and control

1 Introduction

Technological innovation of Industry 4.0 such as horizontal and vertical connectivity and the increasing degree of automation result in a rising complexity of machines, production systems and especially the infrastructures within production systems. In particular, challenges and opportunities arise for companies in the maintenance of production components. As an increasing complexity and dynamics within production make it difficult or even prevent human production engineers to determine the best maintenance schedule, condition monitoring systems have been broadly established to support the decision makers. Machine learning approaches such as predictive maintenance applications build the analytical foundation to predict future machine breakdown [1]. These maintenance strategies address the required flexibility and offer advantages compared to traditional maintenance activities in terms of adaptability. This paper investigates the enhancement and optimization of predictive maintenance management by determining a point in time before a breakdown occurs with a low system load and correspondingly low opportunity cost of the maintenance measure, rather than the last possible time before a breakdown. As a result, both the potential of predictive maintenance is utilized and new cost-cutting opportunity is generated through a higher dynamic system. Due to their immanent flexibility and performance in learning strategies for real (or simulated) systems, reinforcement learning algorithms are suitable to find optimal maintenance schedules in the context of a stochastic production environment. Hence, their applicability is evaluated within this paper.

In previous research, flow line production systems were mainly analyzed by stochastic decision models, whereby the defect of one machine generated a maintenance window for another machine [2]. The approach presented in this paper evaluates the more difficult prediction of optimal maintenance times for independent machines in parallel production systems (Fig. 1) using reinforcement learning. Hence, the proposed method offers a new perspective on opportunistic

✉ Andreas Kuhnle
andreas.kuhnle@kit.edu

Johannes Jakubik
johannes.jakubik@yahoo.com

[1] wbk, Institute of Production Science Karlsruhe Institute of Technology (KIT), Kaiserstrasse 12, 76131 Karlsruhe, Germany

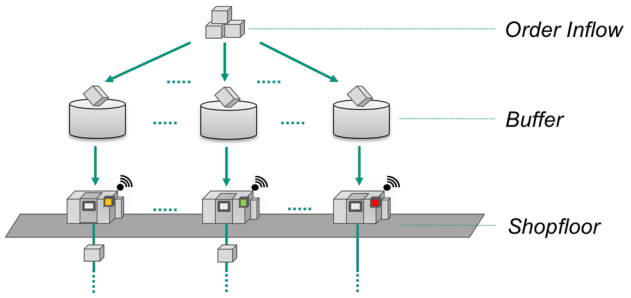

Fig. 1 Structure of a parallel production system consisting of identical machines with upstream buffers

maintenance in terms of the occurrence of opportunistic time windows for maintenance measures.

The underlying implementation is based on the formulation of a decision model extending a predictive model for the optimal execution time of a maintenance action with the aim of minimizing maintenance costs, maintaining production capacities and maximizing the production output. Section 2 covers the fundamentals and literature review of existing approaches, Sect. 3 outlines the production system that is considered in this paper, Sect. 4 describes the reinforcement-learning-based learning agent and, finally, Sect. 5 summarizes the results that are obtained.

2 Fundamentals and literature review on opportunistic maintenance

Currently, the relevance of predictive maintenance in manufacturing processes increases due to its ability to predict upcoming faults and thereby resulting in decreasing downtimes and planning effort for maintenance activities. Different predictive maintenance approaches are summarized in [3]. However, there exist ideas to evolve predictive maintenance and enhance its impact on the production as presented in [4] to combine predictive maintenance with continuous quality control. Another possible evolution of predictive maintenance is opportunistic maintenance, which does not only consider the breakdown times of production components but also takes opportunity costs of the maintenance into account. Recent research by [2] developed stochastic decision model based optimization strategies for the maintenance management of flow line systems. Furthermore [5] analytically proved the cost efficiency of an opportunistic maintenance strategy in a production environment with random waiting times.

The relevance of reinforcement learning for the determination of optimal policies to different manufacturing tasks increases highly as described in [1, 6]. Its applicability in maintenance issues was already proven for both, predictive as well as flow line opportunistic maintenance according to [7, 8].

Wang [8] showed the optimization potentials of reinforcement learning based multi-agent systems for flow line productions as already regarded by [2, 5]. In contrast, the herein presented approach considers reinforcement learning for the maintenance management of machines without interdependencies in a parallel production system. Therefore, the goal is to reduce the interference between production and maintenance through a better synchronization. Thus both the automation in terms of recommendations of maintenance measures can be further increased and a cost reduction by decreasing opportunity costs is achieved.

The aim of the approach presented below is to implement a generic, virtual production system in which the independent machines are observed by condition monitoring. In case of a predicted disturbance, a machine should be maintained at the optimal time, characterized by low machine utilization of the production component, to reduce the opportunity costs of maintenance (Fig. 2). For this purpose, an agent should be able to predict a breakdown implicitly based on the condition monitoring parameters, to control the operation mode of the machine and to carry out the maintenance at the optimal time, by understanding the stochastic correlations.

Therefore a real production system was abstracted and implemented as simulation model to achieve a direct interaction between the agent and the production system environment (see Sect. 3). The virtual system is optimized in

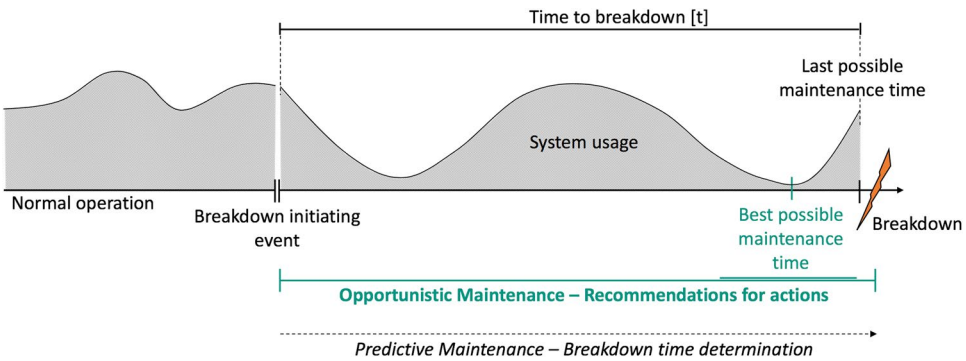

Fig. 2 Opportunistic maintenance for independent machines in a parallel production system regarding the systems usage

terms of maintenance by reinforcement learning agents, as introduced by [9]. The goal of an agent is learning an optimal strategy by taking actions A_t leading from the current state S_t to a new state S_{t+1}. Depending on the desirability of the action (or the state-action-combination) the agent is rewarded by R_{t+1}. Other applications have already shown that multi-agent systems can perform better than a single agent or multiple agents sharing a single knowledge base [10]. This could be proved for this scenario, in which each machine represents the environment for an independent agent, as well (Fig. 3). Hence, in the following sections only the favourable approach of multi-agents will be further analyzed.

The computational results in Sect. 5 are based on the evaluation of four advanced reinforcement learning algorithms out of the very potent *tensorforce* library (which is built on top of *tensorflow*). To reduce the implementation effort, four algorithms were pre-selected by qualitative criteria like performance in suitable benchmark cases as well as heterogeneity of the implementations. The structure of these four algorithms was then analyzed to detect the best theoretical suitability: According to the definition of general strategy gradient methods, their core advantage is their applicability within high-dimensional state and action spaces. Furthermore, these methods are able to learn in stochastic environments. Both factors are relevant within the presented scenario, and therefore implies a disadvantage for *Deep-Q-Networks* (DQN) [11]. In contrast to the *Vanilla Policy Gradient* (VPG) [12] method, *Proximal Policy Optimization* (PPO) and *Trust Region Policy Optimization* (TRPO) as established by [13] are able to optimize non-linear strategies. Thus, these two methods should outperform DQN and the VPG method [13]. In addition, both TRPO and PPO agents come up with improved convergence behaviour, which may result in local rather than global maxima. The difference in performance between achieving and not attaining a global optimum is quantitatively determined in Sect. 5.2. In a prior validation process, it could be shown that PPO agents perform better than DQN and VPG. Furthermore, PPO agents should outperform TRPO agents by using multiple periods of gradient increase for a single policy update as given by [14]. The evaluation of the algorithms' performance with identical parametrization proved the theoretical suitability.

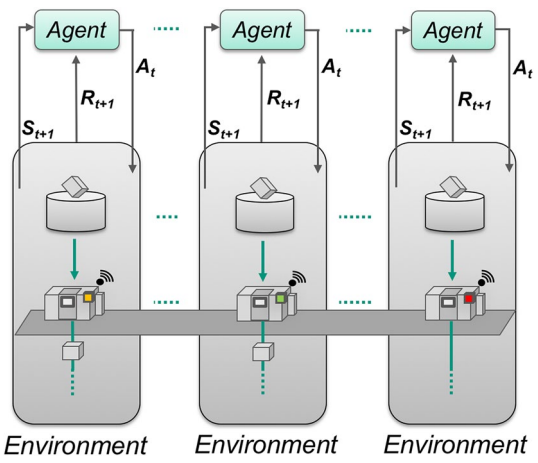

Fig. 3 Setup of a multi-agent system with one agent per machine

3 Production system simulation model

The herein implemented simulation model of a generic production system consists of a variable number of identical, parallel working machines, each with an upstream buffer, which receives a varying number of identical orders at each time step and passes the foremost order to the machine (Fig. 1). The varying number of new orders for each buffer is a random value, created by a rayleigh distribution (which is a distribution, whos right-skewed density has values unequal to zero only in the first quadrant). In the presented simulation, the distribution was parameterized with a maximum of the density at $x = argmax\, f(x) = 0.6$. The machine processes the incoming jobs in a stochastic time period. Afterwards, the job leaves the system. The condition of each machine is monitored and influences the behavior of the machine with regard to a breakdown. Each machine may fail, with the time of the next breakdown depending on the current machine parameters. If a critical, breakdown-initiating value (e.g. speed, torque, temperature) indicating a starting fatigue of materials is observed [15], a malfunction, i.e. unstable operation mode, begins that ends with the breakdown after a certain period of time (again depending on the machine parameters). If one machine fails, it is repaired in a constant time period by a single maintenance engineer who is responsible for every machine in the production system. After a repair, the machine is back in the desired state and the maintenance worker can return to other machines.

3.1 Machine states

There are several states for each simulated machine (see Fig. 4). A machine in normal operation processes orders at normal speed until a breakdown-initiating event occurs. Upon such an event, its internal state changes to a state of normal operating speed in which jobs are processed until the breakdown or a maintenance action ("Normal Operation $_{\text{Until Breakdown}}$").

The further behavior depends on the modes that are considered as comparison and benchmark for the later presented reinforcement learning approach:

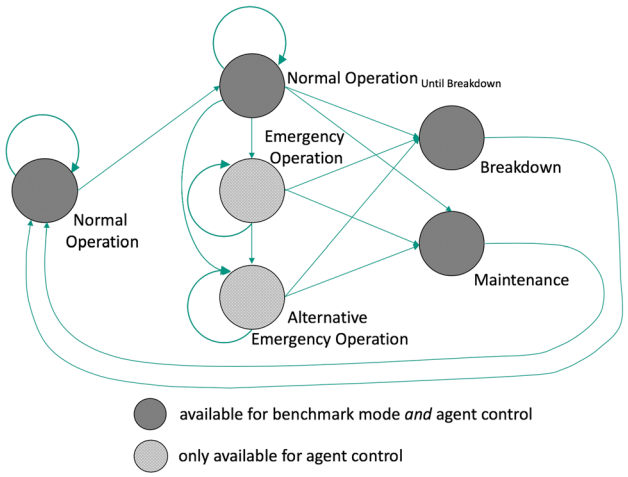

Fig. 4 States and transitions of a machine within the production system

- In the *reactive mode*, a machine works until its defect and is then restored to its desired state by a fixed-duration repair.
- In the *time-based mode*, a machines operates a fixed period of time during the malfunction and each machine is serviced prior to the breakdown (after 60% of the critical period has elapsed) and accordingly returned to the desired condition.
- In the *perfect prediction mode*, a heuristic, with omniscience of faults and defects within the system, always executes maintenance measures shortly before the defect, as it would be performed by predictive maintenance. Although it does not take into account opportunity costs such as the current system utilization, it forms a reliable upper bound, which can only marginally underestimate the optimal solution.

In the *time-based mode* as well as in the *perfect prediction mode*, it is assumed that a maintenance action lasts 30% of the time a repair takes.

3.2 Stochastic modelling of machines' breakdown initiation

The implementation of the model is mainly based on assumptions and the *states* described in Sect. 3.1. Within the simulated time, orders are stochastically added in each iteration. Next, a new currently measured parameter is added to the state representation and the current state of the machine is queried. Based on the current state representation of the system parameters, their variance is calculated and compared with the critical value. If there is only a small deviation from the mean value, the machine remains in the normal state. However, if there is a deviation greater than a critical value, this event is interpreted as a breakdown-initiation. Then, using a bathtub curve, as a model of the breakdown probability, consisting of the combination of two Weibull density functions [16] according to Fig. 5, the breakdown time is calculated in each iteration as follows.

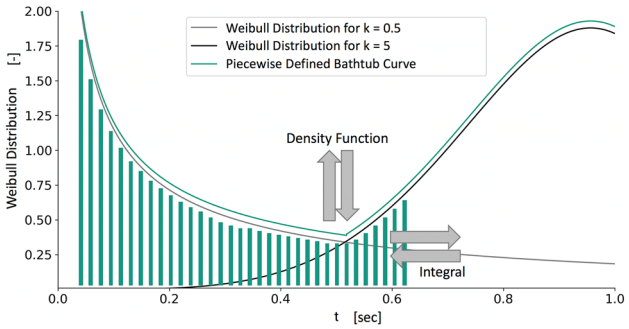

Fig. 5 Determination of the breakdown time using integration of a bathtub curve consisting of two piecewise defined Weibull distributions

The deviations of the machine parameters from the expected value are represented by the corrected variance σ^2. This is normalized as $\frac{1}{|1-\sigma^2|}$ and then multiplied with the Weibull density functions for scaling (Eq. 1). After scaling, the time to breakdown is calculated by integrating the scaled bathtub curve, whereby the integral over the density functions indicates the probability of a breakdown. Then the *t-value*, for which the integral value equals 1 (100%) is determined to stochastically set the time to breakdown based on the Weibull Distribution (Eq. 2).

The advantage of the developed method is the scaling of the combined probability density (Fig. 5). In this way, the consideration that larger deviations from the normal state lead to an earlier breakdown can be represented.

$$f(t) = \begin{cases} \frac{1}{|1-\sigma^2|}\lambda k(\lambda t)^{k-1}e^{-(\lambda t)^k} & \text{t} \leq 0.5 \text{ with k } = 0.5, \\ \frac{1}{|1-\sigma^2|}\lambda m(\lambda t)^{m-1}e^{-(\lambda t)^m} & \text{t} \geq 0.5 \text{ with m } = 5 \end{cases} \tag{1}$$

$$\int_0^t f(t)\,dt \overset{!}{=} 1 \Leftrightarrow t = \cdots . \tag{2}$$

4 Agent implementation

The learning agents enables the automated control of all maintenance in the production system and the agents can learn the optimal maintenance strategy.

The implementation assumes that the production speed of a machine can be reduced. Based on this assumption the controllable states of the simulation are expanded by two

different emergency operations in order to have a more realistic setting (Fig. 4). The emergency operation enables the agent to delay the imminent breakdown to find the best possible time for a maintenance action. In this emergency operation, the production speed is reduced, whereby the time to breakdown is delayed disproportionately longer. In contrast, the second alternative emergency operation is implemented to test and validate the learning agent. Although the production speed decreases in this situation, the time to breakdown of the machine does not change. Accordingly, this operation never pays off and is only implemented to prove the applicability of the agent even in more complex (noisy) systems. The agent monitors changes in production parameters over time and responds at each time step to the current state with an action that results in a transition to a (new) state (Fig. 6).

When implementing opportunistic agents, much effort has to be put into the definition of a suitable interface and the synchronization of simulation and learning system. With regard to the agent a detailed view on the reward signal is essential:

- The agent decides in *normal operation* every time a new parameter is measured, if it considers the machines condition still ordinary or if the condition is critical. It is exclusively positive rewarded, if it is right. A correct decision means choosing the normal operation if the machine parameters are not critical and switching to $normal\ operation_{until\ breakdown}$ if the parameters exceed a critical value. If the agent takes a wrong decision the reward equals zero, which is a sort of a punishment in comparison to the positive reward. This is how the agent learns the *breakdown initiation*.

Subsequently, the agent intentionally induces state transitions through actions. In each state, the agent is therefore asked after every processed order whether it wants to change the state according to the directed graph in Fig. 4:

- If an action is *invalid*, i.e. the state is not allowed, the agent will *not* receive any reward.
- The *alternative emergency operation* to disturb the agent is never rational and therefore neither rewarded.
- Valid state transitions between normal $operation_{\text{until breakdown}}$ and the two different emergency operations are, considered with a positive reward equal to edge weights in directed graphs. The amount of reward for the transition from one operation mode to another depends on the current buffer volume. For example, if the buffer volume is high, it is better for the agent to choose the emergency operation because it delays the breakdown and waits for a time with lower opportunity cost (i.e. a smaller buffer volume) to perform a maintenance action.
- If the agent does not decide to initiate a maintenance action shortly before a breakdown, it will be penalized with a negative reward of −1.

The key of the agent's reward is the behaviour of initiating a maintenance action. The reward depends on a current buffer volume τ and the time t until breakdown:

$$R_t(\tau) = \begin{cases} \frac{1}{ln(t)} & \text{if } \tau = 0 \quad \text{and} \quad t \geq 1 \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

The condition $\tau = 0$ is due to a relative small maximum buffer volume of ten orders in the buffer during the simulation. During different parametrization of this condition, it could be observed that "softer" constraints irritate the agent and reduce learning performance. The choice of the natural logarithm leads to a reasonable scaling of the comparatively large values t and ensures continuity.

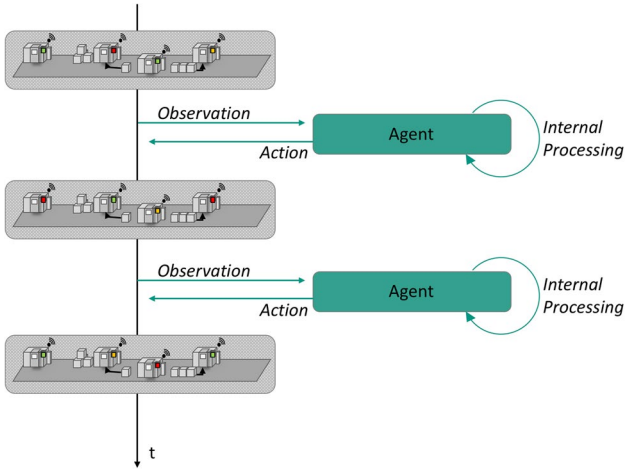

Fig. 6 Interaction between agent and environment

5 Computational results

The computational results are structured as follows: The relevant learning objectives and their achievement are discussed first. Then a comparison of the processed orders of the intelligent controlled production with the heuristically determined processed orders based on traditional maintenance methods follows (e.g. reactive, time-based). Based on this, a cost comparison is carried out which unambiguously quantifies the potential of the implemented method. Finally, a statistical analysis of the results is given in Sect. 5.2 with a sample from the scope $n = 40$ to analyse the robustness of the learning based approach to reach a global optimal solution.

5.1 Achieving successful learning performance

The primary objective of the underlying agent implementation is to initiate a maintenance action in the period just before a defect. Achieving this goal is illustrated in Figs. 7 and 8 by the performance improvement during the learning phase starting from an initial state with a totally random behaviour. Figure 7 illustrates the remaining time to breakdown which is measured every time a maintenance is initiated by the agent during the critical state. This remaining time to breakdown is identified as unused potential, which decreases over the learning process. From about 20% of the simulated time, the agent follows a strategy that brings the maintenance action closer to the defect. Hence, the agent proves that it is able to implicitly learn the prediction of a breakdown without being directly rewarded for it. Based on this, the agent performs a preventive action shortly before a defect. As shown in Fig. 9, there are still breakdowns that were not prevented by the agent, but these breakdowns occur mostly just at the beginning of the learning process. This is illustrated by the agent receiving mainly rewards greater than or equal to zero, whereby the negative reward of −1 indicates the occurrence of a not prevented defect. Accordingly, the agent learns to prevent breakdowns by preventive maintenance actions shortly before the defect occurs. However, for each maintenance action, the agent does not only seek a point in time as close as possible to a breakdown, but rather a combination of a point in time as late as possible and a point in time with the least possible buffer volume, i.e. optimal from an opportunistic point of view. This behavior is illustrated in Fig. 8 by the buffer volume at the time of maintenance action, where the decrease in buffer volume at the time of breakdown proves the effective learning.

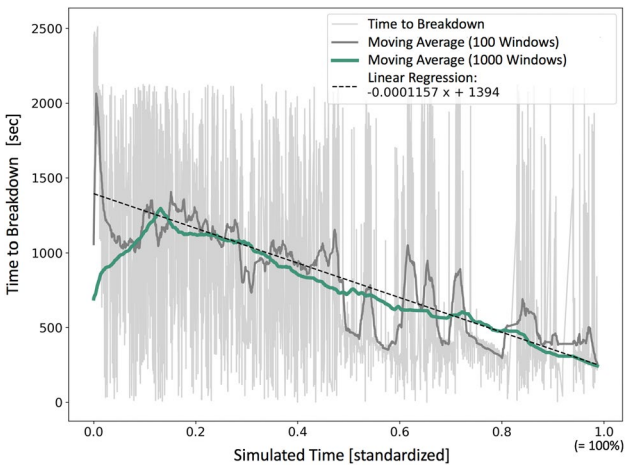

Fig. 7 Remaining time to breakdown at the time of performing a maintenance action

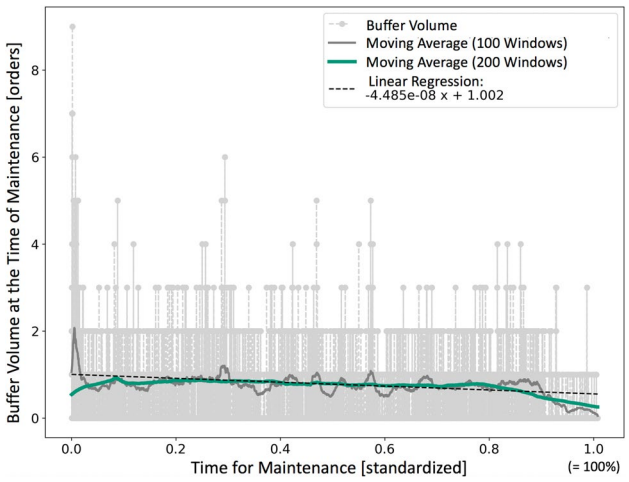

Fig. 8 Buffer volume at the time of performing a maintenance action

The combination of these results leads to the conclusion that the agent learns the desired behavior by observing the impact of its actions and thereby optimizing the overall maintenance management. In addition to that, the implementation also proves that noisy data in the form of meaningless states does not prevent the learning process of the agent (see Fig. 10). Furthermore, the agent is able to learn the validity of the actions depending on the current state. During the learning process, a total of around 14,000 invalid actions are selected, amounting to 460,000 actions (in total roughly 3%), with about a quarter of the non-valid actions taking place in the very first learning period. This equals the exploration phase until around 10% of simulated time. Thereafter, the slope of the curve decreases significantly and hardly any invalid action is selected by the agent. That also demonstrates that edge weights in a directed graph represent traceable rewards for an agent.

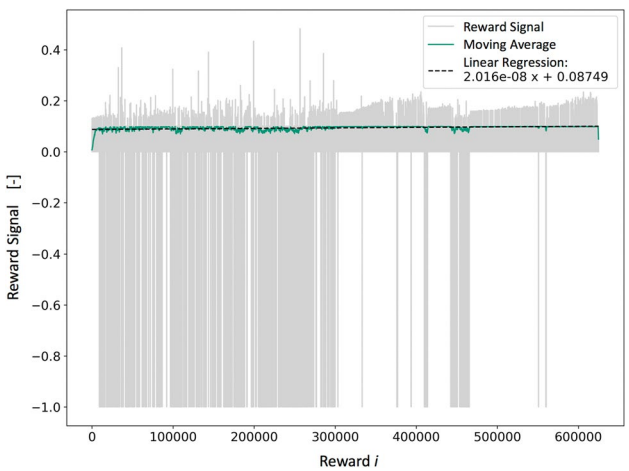

Fig. 9 Reward signal of an opportunistic PPO agent

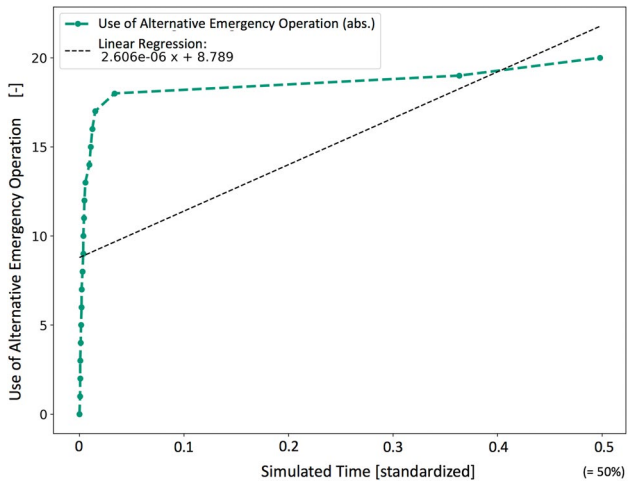

Fig. 10 Use of the alternative emergency operation

The relevance and significance of the completed learning process is illustrated by a performance and cost comparison in the following section.

5.2 Economic evaluation of output and costs

The potential of the proposed PPO multi-agent system is quantified in Fig. 11 by comparing it with the benchmarks reactive, time-based and perfect prediction introduced in Sect. 3.1. Time-based maintenance means carrying out a maintenance measure after 60% of the critical period has elapsed. Comparing with this maintenance strategy the agent hardly differs at the beginning, only after about 25% of the simulated time.

In this first section, the agent and the perfect prediction, regarded as upper bound, have different gradients, but in the subsequent phases both functions are approximately parallel, suggesting that the learning behaviour is optimal. To further investigate this, Fig. 12 compares the gradients of the perfect prediction mode and the agent-based control. The differences of the gradients are determined for each time step and their moving averages are displayed. As previously described, at the beginning, up to about 5% of the simulated time, the gradient of the perfect prediction mode is significantly higher than the gradient of the agent controlled maintenance. In the following phase up to about 50% simulation time, the gradient difference alternates by about 20%. In the optimal phase of learning (see Fig. 7) after about 80% simulation time, the gradient of the

Fig. 11 Comparison of intelligent maintenance with reactive and preventive maintenance as well as the perfect prediction

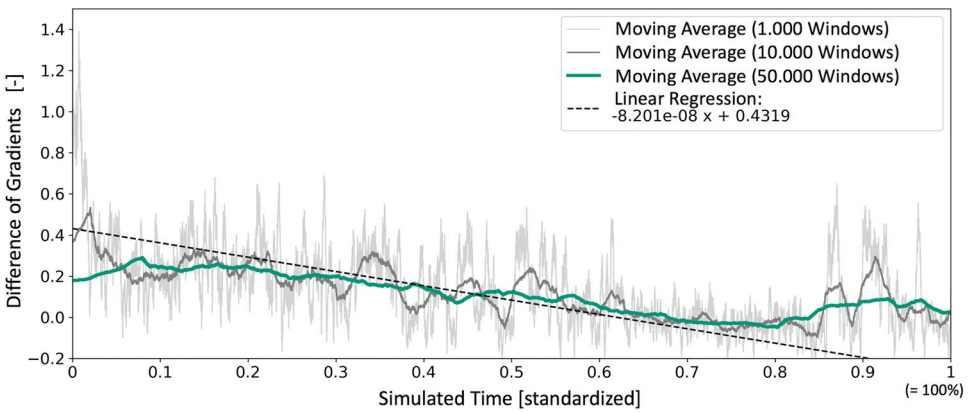

Fig. 12 Difference of gradients

perfect prediction mode and the agent control actually differs only marginally. In this phase, the difference is also negative, which means that the agent controlled system processes more orders than the perfect prediction benchmark, which is due to idle times that the opportunistic agent uses more efficiently for maintenance (as per assumption in Sect. 3.1).

This shows that an opportunistic agent is able to outperform existing benchmark maintenance approaches. Moreover, it could be shown both that the agent is able to learn the optimal strategy. Hence, the results are a completely new form of agent-controlled opportunistic and additionally maintenance.

In runs where the learning-based agent does not find the optimal solution due to the underlying stochastic, the absolute deviation of the number of processed jobs from the best observed value is around 15% demonstrating consistency and robustness. On the other hand, the worst measured result during the validation process is still better than the outputs of reactive and time-based maintenance. For better quantification of the stochastic results of the agent control a sample of $n = 40$ is used to evaluate the overall results (Fig. 13). The density function is based on a histogram, whereby the function itself represents an approximation of the normal distribution (central limit theorem) that is based on the sample values. The area of high relevance is $[+\sigma, +2\sigma]$ or $[+\sigma, +\infty)$. Accordingly to the normal distribution assumption, the probability of obtaining values in the range $[+\sigma, +\infty)$ is around 15.87%.

These results focus on the increasing output that is achieved with the implementation as a result of reducing the opportunity cost of the maintenance strategies and reducing production downtime due to a lower total number of maintenance activities. On the other hand, the opportunistic maintenance reduces the costs of breakdown prevention significantly by targeted measures at the latest possible times, creating an additional financial potential. Executing maintenance measures as late as possible (see Fig. 7) means a reduction in costs, as fewer maintenance operations have to be carried out. The results are summarized in Table 1.

This does not take into account the more advanced and thus better exploited wear rate of the machine components, which is why the actual value tends to be underestimated. In addition, the potential is even greater under a long term consideration as the divergence between the approaches then continues to increase. The agent's first phase of orientation and exploration negatively impacts the number of defects and maintenance results as shown in Figs. 7 and 9.

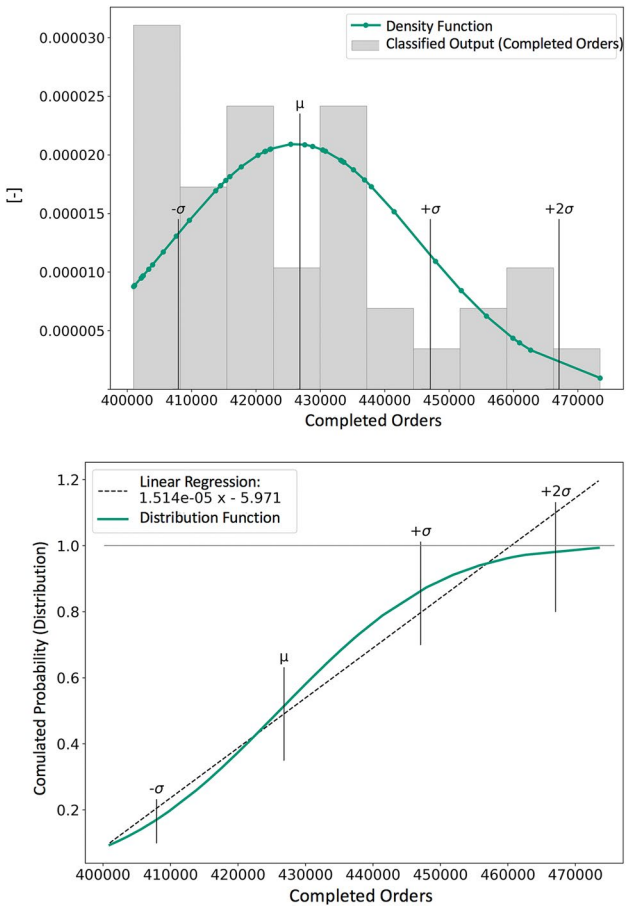

Fig. 13 Density and distribution function of a sample of simulation results of $n = 40$ in terms of processed jobs

6 Conclusion and outlook

The herein presented approach shows a novel opportunistic maintenance strategy and gives further evidence of the successful application of reinforcement learning algorithms in manufacturing. It could be shown that the implemented opportunistic, multi-agent controlled maintenance results in an optimized system output and a significant reduction of both maintenance costs and effort. Furthermore, the performance is robust and could be even improved with an isolated implementation without meaningless states and with knowledge whether actions are valid. Such information can be implied in future implementations for more stability in the learning process and for an increased performance. Hence, the introduced approach offers still optimization potential in terms of the learning performance. The approach could be also extended to take additional tasks of production control into account such as order dispatching and scheduling.

Table 1 Comparison of economically relevant factors regarding different maintenance strategies of an independent machine within the production system (values rounded)

	Agent	Reactive maintenance	Time-based maintenance	Perfect prediction mode
Completed jobs	460.000	330.000 (-28%)	395.000 (− 14%)	500.000 (+ 9%)
Breakdowns	570	2.950 (+420%)	✗	✗
Maintenance measures	3.700	✗	6.500 (+120%)	4.550 (+55%)
Exemplary costs[a]	1.680	2.950 (+76%)	1.950 (+16%)	1.365 (− 19%)

[a]Total sum of costs of maintenance and repairs in units of cost of a repair. The cost of a maintenance measure is assumed to be 30% of the cost of a repair, as a maintenance execution linearly takes 30% the time of a repair

Acknowledgements We extend our sincere thanks to the German Federal Ministry of Education and Research (BMBF) for supporting this research project 02K16C082 Produktionsbezogene Dienstleistungssysteme auf Basis von Big-Data-Analysen (ProData).

References

1. Wuest T (2016) Machine learning in manufacturing: advantages, challenges, and applications. Prod Manuf Res 4:23-45
2. Colledani M, Magnanini MC, Tolio T (2018) Impact of opportunistic maintenance on manufacturing system performance. CIRP Ann 67(1):499–502
3. Hashemian HM, Bean Wendell C (2011) State-of-the-art predictive maintenance techniques. IEEE Trans Instrum Meas 60(10):3480–3492
4. Lindström J, Larsson H, Jonsson M, Leyon E (2017) Towards intelligent and sustainable production: combining and integratingonline predictive maintenance and continuous quality control. Procedia CIRP 63:443–448
5. Yang L et al (2018) Opportunistic maintenance of production systems subject to random wait time and multiple control limits. J Manuf Syst 47:12–34
6. Stricker N et al (2018) Reinforcement learning for adaptive order dispatching in the semiconductor industry. CIRP Ann 67(1):511–514
7. Wang X et al (2014) Reinforcement learning based predictive maintenance for a machine with multiple deteriorating yield levels. J Comput Inf Syst 10(1):9–19
8. Wang J (2016) Multi-agent reinforcement learning based maintenance policy for a resource constrained flow line system. J Intell Manuf 27(2):325–333
9. Sutton RS, Barto AG (2017) Reinforcement learning: an introduction. MIT Press, Cambridge
10. Crites RH, Barto AG (1995) Improving elevator performance using reinforcement-learning. Adv Neural Inf Process Syst 8:1017–1023
11. Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing atari with deep reinforcement learning. CoRR. arXiv:abs/1312.5602
12. Williams RJ (1992) Simple statistic gradient-following algorithms for connectionist reinforcement learning. Mach Learn 8:229–256
13. Schulman J, Levine S, Abbeel P, Jordan M, Moritz P (2015) Trust region policy optimization. In: Proceedings of the 31st International Conference on Machine Learning, vol 37, pp 1889–1897
14. Schulman J (2017) Proximal policy optimization algorithms. Adv Neural Inf Process Syst 8:1017–1023
15. Schijve J (2009) Fatigue of structures and materials. Springer, Amsterdam, pp 15–21
16. Xie M, Lai CD (1996) Reliability analysis using an additive Weibull model with bathtub-shaped failure rate function. Reliab Eng Syst Saf 52(1):87–93