Optimal Part Flow In Maintenance Service Contracts Of Gas Turbines

Luca Bellani

Aramis s.r.l., Milano, Italy. E-mail: luca.bellani@aramis3d.com

Michele Compare

Aramis s.r.l., Milano, Italy. E-mail: michele.compare@aramis3d.com
Energy Department, Politecnico di Milano, Italy

Enrico Zio

Aramis s.r.l., Milano, Italy
Energy Department, Politecnico di Milano, Italy. E-mail: enrico.zio@polimi.it
MINES ParisTech, PSL Research University, CRC, Sophia Antipolis, France
Eminent Scholar, Department of Nuclear Engineering, College of Engineering, Kyung Hee University, Republic of Korea

Marzia Sepe

Baker Hughes, Florence, Italy, E-mail: Marzia.Sepe@bhge.com

Francesco Annunziata

Baker Hughes, Florence, Italy, E-mail: Francesco.Annunziata@bhge.com

Fausto Carlevaro

Baker Hughes, Florence, Italy, E-mail: Fausto.Carlevaro@bhge.com

IGT manufacturers offer maintenance service contracts that guarantee specific production rates. At every maintenance action (i.e., corrective or scheduled), capital parts are removed from the GTs and repaired at workshop, unless they are scrapped because they have reached their pre-fixed maximum number of working hours. The repaired parts are put back at the warehouse for future use. The parts removed from the GTs are replaced by parts newly purchased or taken from the warehouse. This maintenance policy entails a part flow, which is managed through decisions on both the removed parts (repair or scrap?) and the parts to be installed on the GT (parts new or taken from the warehouse?). Such decisions strongly impact the profitability of the maintenance service contract and depend on many variables, e.g. remaining time up to the end of the contract, availability of spares, costs related to the repair and purchase actions, etc. Furthermore, in the dynamic of the part flow, every decision conditions the successive ones, as it modifies the warehouse composition. We formalize the part flow problem as a Sequential Decision Problem and solve it by both integer linear programming framework and reinforcement learning, taking as reference a scaled-down case study derived from industrial practice. Final considerations are drawn about both approaches.

Keywords: Sequential Decision Making, Logistic Optimization, Gas Turbines.

1. Introduction

Gas Turbines (GTs) employed in the Oil&Gas industry are made up of expensive capital parts. To avoid forced outages, which cause business interruptions with severe economic losses, GTs are periodically maintained. The management of the flow of the GT parts undergoing maintenance is a major issue for the profitability of the operation of the GTs: at the end of every maintenance cycle, the capital parts are removed from the GTs and replaced by parts of the same type available at the warehouse Boyce (2012). The removed parts are scrapped if they have reached their pre-fixed maximum number of working cycles or, otherwise, they are repaired at the workshop and put back in the warehouse, ready to be installed at the next Maintenance Shutdowns (MSs) of another GT in the same Oil & Gas plant. Thus, at every periodic MS, a decision has to be made on both the removed part (send it to the workshop for repair or scrap it?) and the part to be installed on the GT (new part or taken from the warehouse?).

Proceedings of the 30th European Safety and Reliability Conference and the 15th Probabilistic Safety Assessment and Management Conference
Edited by Piero Baraldi, Francesco Di Maio and Enrico Zio
Published by Research Publishing, Singapore.
ISBN: 978-981-14-8593-0; doi:10.3850/978-981-14-8593-0

To take these decisions, Decision Makers (DMs) have to consider that, on the one hand, the cost of the repair actions and the risk of forced outage due to the failures of the capital parts increase with part age; thus, actions favoring the scrapping of old parts are beneficial for risk and workshop costs. On the other hand, scrapping old parts entails purchasing new capital parts, whose costs are typically larger than those of the repair actions. Obviously, the parts installed on the GTs will no longer be available at the warehouse for the next MS and even if they are not scrapped, they return to the warehouse with a reduced number of remaining working cycles. Thus, the decision at every MS modifies the decisions at the next MSs. In this sense, the part flow management can be framed as a Sequential Decision Problem (SDP) Sutton and Barto (2018).

From above, it clearly appears that part-flow management is a complex issue, where the DM has to seek for the least expensive sequence of maintenance decisions over the GT operation time horizon (i.e., the optimal policy), rather than greedy decisions with the smallest immediate cost. To do this, the DM has to consider many variables such as the availability of spares, the remaining time up to the end of the GT operation horizon, the costs related to the repair and purchase actions, etc.

Moreover, GTs are normally expected to work for a very long time, e.g., up to 25 years Boyce (2012). This requires a thorough cost analysis to take into account the effect of the time value of the money through the discount rate of future expenditures, which leads to prefer part flow policies postponing as long as possible the most expensive actions Gollier (2002).

To the authors' best knowledge, despite the relevance of part flow management for the Oil&Gas industry, systemic approaches to address it are lacking and GT maintenance DMs still rely on experience-based rules such as Most Residual Cycles (MRC): the removed parts are always repaired until the end of the GT time horizon and the part with the largest residual life among those available at the warehouse is installed on the GT; new parts are purchased and installed on the GTs only when the warehouse is empty. This simple and intuitive rule guarantees the smallest repair cost at the smallest probability of failure and it delays the costly actions as long as possible. The aim of this work is to formalize the part flow as a SDP problem. This is solved by both integer linear programming (ILP, Schrijver (1986); Bertsimas and Tsitsiklis (1997)) and Reinforcement Learning, taking as reference a scaled-down case study derived from industrial practice. The application of both these algorithms allows obtaining the same global optimal solution, which is compared with that of the MRC policy to show that this latter is not optimal. This highlights the need to treat the part flow management through a systemic approach. A comparison of ILP and RL is also outlined.

The paper is organized as follows: in Section 2, the problem setting is presented. The ILP optimization model is presented in Section 3, whereas the corresponding RL model is presented in Section 4. In Section 5, a realistic case study is introduced to show the non-optimality of the MRC policy. Finally, conclusions are drawn in Section 6.

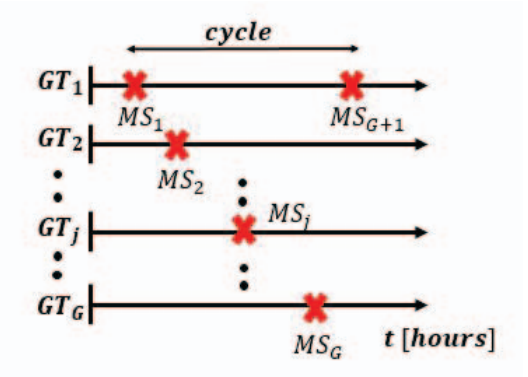

Fig. 1.: Part of contract's story

2. Model setting

Assume that in a Oil & Gas plant a number G of GTs are operated and periodically maintained. For simplicity, we assume that every GT contains a single capital part, only, this assumption having no impact on the generality of the proposed framework, as the part flows of the capital parts are independent on each other and, thus, they can be optimized separately.

We also assume that the maintenance staggering is such that two MSs are never done simultaneously and that the repair time is negligible with respect to the time between two MSs. This entails that the parts removed and then repaired at any MS are available at the next MS.

Finally, the Residual Useful Life (RUL) of the parts indicates the r remaining working cycles; then, the RUL values range between $r = 0$, in case of parts that must be scrapped, and $r = R$, in case of new parts.

The number of remaining scheduled MSs for every GT is Z and, thus, a total number $T = Z \cdot G$ of MSs will be performed during the GT plant time horizon.

Under these assumptions, the GT undergoing maintenance at the k-th MS, $k \in \{1, \ldots, T\}$, is univocally identified by:

$$g = k - \lfloor (k-1)/G \rfloor \cdot G \tag{1}$$

where $\lfloor \circ \rfloor$ is the integer part of its argument $\circ$. Given this relationship, from now on the GTs are indicated by the order of occurrence of their first MS (Figure 1).
At the k-th MS, the plant manager has to take the following decisions:

- For the part removed from the g-th GT, decide whether to repair or scrap it. $C^{rep}(r)$ is the cost of repairing a part with $r \in \{1, .., R\}$ remaining cycles, whereas C^{scrap} is the cost of scrapping a part.
- To replace the removed part, decide whether to buy a new part or select one from those available at the warehouse, if any. C^{pur} is the cost of purchasing a new part, whereas the cost of selecting a part from the warehouse is zero, as the repair costs have already been accounted for.

To model the constraints in the part flow management, we introduce the integer variable $w_{r,k}$ to indicate the number of parts with RUL equal to r available at the warehouse for the k-th MS and the binary variables $d_{r,k}$ such that $d_{r,k} = 1$ indicates that the GT maintained at k-th MS has r remaining cycles. Obviously, at each MS, each gas turbine must be provided with a single part, thus $\sum_{r=1}^{R} d_{r,k} = 1 \; \forall k \in \{1, \ldots, T\}$.

3. ILP framework

At the k-th MS, we indicate by $m_{r,k}$ the binary variable such that $m_{r,k} = 1$ if the removed part with $r \in \{1, \ldots, R\}$ remaining cycles is repaired and $m_{r,k} = 0$, otherwise.
With respect to the replacement of the removed part $\alpha_{r,k}$ is the binary variable such that $\alpha_{r,k} = 1$ if one out of the components with $r \in \{1, \ldots, R\}$ remaining cycles is selected; $\alpha_{r,k} = 0$, otherwise. $\alpha_{0,k}$ is the binary indicator variable such that $\alpha_{0,k} = 1$ if a new component is purchased and $\alpha_{0,k} = 0$, otherwise.
From above, the cost incurred at the kth MS is:

$$C_k = \alpha_{0,k} \cdot C^{pur} + \sum_{r=1}^{R-1} [C^{rep}(r) \cdot m_{r,k} + C^{scrap} \cdot (1 - m_{r,k})] \tag{2}$$

If we further assume that the time value of money is represented by a constant discount factor $\gamma \in (0, 1)$, the present value (PV_k) of the cost C_k incurred during $k-$th MS is

$$PV_k = \gamma^{\frac{t_k - t_0}{T_\gamma}} \cdot C_k$$

where t_k is the time at which the $k-$th MS is performed, t_0 is the time at which the decisions are made and T_γ the time at which the present value is γ times the real value.
The present value of the total maintenance expenditures incurred in the whole time horizon, which is the objective function that the maintenance decision maker wants to minimize, is given by the sum of the PV of all costs incurred.
Within the ILP framework, the problem of finding the optimal part flow can be formulated as follows:

$$\min_{\mathbf{a},\mathbf{m}} \sum_{k=1}^{T} (PV_k) \tag{3}$$

subject to the constraints 4-16, where

- $d^i_{r,k}$ indicates the RUL of the components initially set on the GTs. Namely, $d^i_{r,k} = 1$, if the component on the GT g which is first maintained at $k + G$-th MS has r remaining cycles; $d^i_{r,k} = 0$, otherwise.
- w^i_r indicates the initial composition of the warehouse: w^i_r is the number of components with r remaining cycles initially available in the warehouse.

Equation 3 defines the objective function of the optimization, i.e. minimize the total discounted maintenance costs over the whole life time of the power plant, i.e. the sum of the present value of all costs incurred at each MS.
Equations 4 state that at each MS, exactly one part has to be set on the maintained turbine. Equations 5 constrain the choice of the RUL of the part to be set on the turbine to the current availability of the warehouse. Equations 6 state the multi-period constraints for the warehouse, accounting for the parts which are removed from the turbines at each MS. Notice that, as stated in Equations 7, components that are removed from the GTs have performed at least one cycle, thus the number of their remaining cycles cannot be equal to R: no repair actions can be performed on parts with RUL$= R$. Equations 8 formalize that, at each MS, only parts with $r \geq 1$ remaining cycles can be repaired, i.e. that parts with RUL=0 must be scrapped. Equations 9-10 are the multi-period constraints for the turbines, i.e. they link the action performed at $k-$th MS to the part which is set on the turbine during its MS.
Equations 11-14 define the integer values that can be assumed by the variables in the model. Finally, Equations 15 and 16 initialize the composition of the warehouse (i.e., the initial number of components having r remaining cycles $\forall r \in \{1, \ldots, R\}$) and of the turbines (i.e., the number of remaining cycles of the component which is set on each turbine), respectively. Notice that we used negative times to refer to events that occurred before the first MS (i.e., the parts which were already set on

$$\sum_{r=0}^{R} \alpha_{r,k} = 1 \qquad \forall k \in \{1,\dots,T\} \tag{4}$$

$$\alpha_{r,k} \leq w_{r,k} \qquad \forall k \in \{1,\dots,T\},\ \forall r \in \{1,\dots,R\} \tag{5}$$

$$w_{r,k+1} = w_{r,k} - \alpha_{r,k} + m_{r,k} \qquad \forall k \in \{1,\dots,T\},\ \forall r \in \{1,\dots,R-1\} \tag{6}$$

$$m_{R,k} = 0 \qquad \forall k \in \{1,\dots,T\} \tag{7}$$

$$m_{r,k} \leq d_{r+1,k-G} \qquad \forall k \in \{1,\dots,T\},\ \forall r \in \{1,\dots,R-1\} \tag{8}$$

$$d_{r,k+1} = \alpha_{r,k} \qquad \forall k \in \{1,\dots,T\},\ \forall r \in \{1,\dots,R-1\} \tag{9}$$

$$d_{R,k+1} = \alpha_{R,k} + \alpha_{0,k} \qquad \forall k \in \{1,\dots,T\} \tag{10}$$

$$\alpha_{r,k} \in \{0,1\} \qquad \forall k \in \{1,\dots,T\},\ \forall r \in \{0,\dots,R\} \tag{11}$$

$$m_{r,k} \in \{0,1\} \qquad \forall k \in \{1,\dots,T\},\ \forall r \in \{1,\dots,R-1\} \tag{12}$$

$$w_{r,k} \in \{0,\dots,W\} \qquad \forall k \in \{1,\dots,T+1\},\ \forall r \in \{1,\dots,R\} \tag{13}$$

$$d_{r,k} \in \{0,1\} \qquad \forall k \in \{1,\dots,T+1\},\ \forall r \in \{1,\dots,R\} \tag{14}$$

$$d_{r,k} = d^i_{r,k} \qquad \forall r \in \{1,\dots,R\},\ \forall k \in \{-G+1,\dots,0\} \tag{15}$$

$$w_{r,1} = w^i_r \qquad \forall r \in \{1,\dots,R\} \tag{16}$$

the GTs).
The ILP optimization task has been solved by the command `intlinprog` of MATLAB®, which uses a simplex method in combination with continuous relaxation, integer linear preprocessing Andersen and Andersen (1995), cuts Cornuéjols (2008) and branch and bound algorithms Nemhauser and Wolsey (1999).

4. RL framework

In this Section, we give some details about the model-free RL algorithm here developed for part flow optimization. Generally speaking, RL is based on the idea that the DM, who is usually referred to as agent, learns from his/her interactions with the environment to achieve prefixed goals, without knowledge on the updating dynamics of the environment and the specific effect of his/her actions. Thus, we only need to define the state of the environment, the actions available at each state and the corresponding rewards Sutton and Barto (2018).
The action taken at the k-th MS is indicated as:

$$A_k = \sum_{\rho=0}^{R} (\alpha_{\rho,k} \cdot \rho) + \mu_k \cdot (R+1) \tag{17}$$

where $\mu_k = \sum_{k=1}^{R} m_{r,k}$ is the binary indicator variable such that $\mu_k = 1$ if a repair action is performed on the part removed at MS k and $\mu_k = 0$ otherwise. The choice of not including the RUL of the part removed from the GT in the action definition is due to the fact that this does not depend on the decision taken by the DM. Notice that within this framework, Equations 6, 8 change to 18 and 19 respectively:

$$w_{r,k+1} = w_{r,k} - \alpha_{r,k} + \mu_k \cdot d_{r+1,k-G} \tag{18}$$

$$\mu_k \leq \sum_{r=2}^{R} d_{r,k-G} \tag{19}$$

The state at the k-th MS is defined by the vector $\mathbf{S}_k \in \mathbb{N}^{R+1}$, $k \in \{1,\dots,T\}$, whose j-th element is:

$$S_{k,j} = \begin{cases} w_{j,k} & \text{if} \quad j \in \{1,\dots,R\} \\ k & \text{if} \quad j = R+1 \end{cases} \tag{20}$$

In words, the first R entries of the state vector at the k-th MS define the number of parts with the different RUL values available at the warehouse, whereas the last entry updates the number of MSs performed Li et al. (2017). Then, the total number of possible states is $T \cdot (W+1)^R$.
Notice that the state vector $\mathbf{S}_k$ does not encode any information about the parts currently installed on the GTs. This leads the SDP to not fully satisfy the Markov property Sutton and Barto (2018), Whitehead and Lin (1995), which requires that the knowledge of the current state of the environment be sufficient to predict its future evolution. To see that this property is here infringed, we can notice from Eq. (18) that the state reached by taking any action is completely defined only if we know $d_{r,\kappa}$ $\forall r \in \{1,\dots,R\}, \forall \kappa \in \{k-G+1,\dots,k\}$ (i.e., if we know the RUL of the part installed on each GT). Since these variables are not encoded in the state vector, we observe that we have transitions towards different states even if we take the same action on the environment in a given state. As

pointed out in Sutton and Barto (2018), the loss of the Markov property typically affects the RL capability of fast convergence to the optimal solution, although RL is eventually able to find it.
The choice of not including the RUL values of the parts installed on the GTs into the state vector has a twofold justification. On one side, including them would broaden the vector state size, which becomes $T \cdot (W+1)^R \cdot R^G$: this leads to heavy computational burdens, undermining the applicability of the proposed framework. For example, if we consider that in a real industrial application $R = 6$ and $G = 10$, then the proposed definition of state would reduce the state vector size by 6^{10}. On the other side, we observe from Eqs. 18 and 9 that the environment state after G MSs is known for any sequence of G actions. Then, the process describing the evolution of the state is a G-order Markov process, in the sense that the knowledge of the sequence of states at the last G events is sufficient to predict its future evolution. Thus, the information about the RUL of the parts installed on the GTs becomes redundant after G steps.
The base reward at the k-th MS is the opposite of the maintenance cost, $-C_k$, as RL is usually framed as a maximization task, whereby minimizing cost is equivalent to maximizing its opposite.
In the RL framework, each state-action pair is described by $Q_\pi(\mathbf{S}_k, A_k)$, which measures the expected return starting from state $\mathbf{S}_k$, taking action A_k and thereafter following policy π Sutton and Barto (2018):

$$Q_\pi(\mathbf{S}_k, A_k) = \mathbb{E}_\pi[\sum_{t=k}^{T} (\gamma^{t-k} \cdot (-C_t))|\mathbf{S}_k, A_k] \tag{21}$$

where $k \in \{1, \dots, T\}$.
In this work, we use the SARSA(λ) algorithm to find the best approximation of the values of $Q_\pi(\mathbf{S}_k, A_k)$, $k = 1, \dots, T$, which simulates a large number of state-action episodes while guaranteeing a faster convergence (e.g., Sutton and Barto (2018); Wang et al. (2013)). The SARSA(λ) algorithm relies the following updating formula in Eq. 22, at every MS, k, where $\lambda \in [0, 1]$ is the parameter governing the eligibility trace and $\alpha_n \in [0, 1]$ is the learning rate at the n-th episode (see Appendix for further mathematical details).
The choice of using SARSA(λ) among the available RL algorithms (e.g., Sutton and Barto (2018); Szepesvári (2010)) is justified by the fact that within the family of value-based RL algorithms, SARSA(λ) has been shown to be a very effective on-policy method (Szepesvári (2010)). This makes it simpler to extend it to the eligibility trace paradigm, which guarantees fast and robust convergence, especially in case of finite time horizon SDPs (Sutton and Barto (2018)). On the contrary, off-policy RL algorithms such as Q(λ) do not allow updates that use all the rewards up to the end of the finite horizon due to the presence of explorative actions.

5. Case study

We consider the case study summarized in Table 1, derived from a real industrial application.
There are $G = 2$ GTs (first column in Table 1), which are maintained for $Z = 10$ cycles, each (second column). The maximum component RUL, R, and the maximum number of available parts in the warehouse for each RUL value, W, are both set to three (third and forth columns in Table 1). The costs are shown in columns 5-8 of Table 1. These values are for illustration, only.
Moreover, we assume that the maintenance staggering is such that the time distance between two consecutive MSs is constant and that the present value of costs decreases by γ at each MS (i.e., $\gamma_{k+1} = \gamma \cdot \gamma_k$, $\gamma_1 = \gamma$). The discount rate is set to $\gamma = 0.99$: on the one hand, if there are two feasible solutions with same undiscounted cost, $\gamma < 1$ leads to select that which postpones the expenditures as long as possible. On the other hand, $\gamma_{T=20} = 0.99^{20} \simeq 0.82$; thus, a purchase action made at the last MS, $T = 20$, entails a cost larger than that of a repair action performed at the first MS ($C_T = 82 > C^{rep}(2) = 50$). Thus, setting $\gamma = 0.99$ guarantees to find the sequence of actions that yield the minimum overall cost and with the largest delays for expenditures.
The application of the MRC rule to the considered case study is summarized in Table 2. Namely, the first column reports the MS counter $k \in \{1, \dots, T = 20\}$. The following three columns define the warehouse composition at the corresponding MS (i.e., $w_{r,k}, k \in \{1, \dots, T = 20\}, r \in \{1, \dots, R = 3\}$). For example, at the beginning of the considered time horizon, i.e., at $k = 1$, there are two parts with $r = 1$ remaining cycle ($w_{1,1} = 2$), one part with $r = 2$ remaining cycles ($w_{2,1} = 1$) and one new part ($w_{3,1} = 1$). The following three columns detail the action taken at the $k-$th MS. For example, at the first MS, the RUL of the part installed on GT $g = 1$ is $r = 3$ (column 7), the removed part with $r = 3 - 1 = 2$ remaining cycles is repaired (i.e., $m_{1,2} = 1$, $\mu_1 = 1$) (column 8) and there is no purchase of new parts (i.e., $a_{1,3} = 1$) (column 9).
Finally, the last two columns report the maintenance cost C_k and its present value $PV_k = C_k \cdot 0.99^{k-1}$), respectively, at MS $k \in \{1, \dots, T = 20\}$. At MS $k = 1$, $C_1 = PV_1 = 50$: a repair action is performed on a part with $r = 2$ remaining cycles, with no purchasing.
From the analysis of Table 2, the first purchase is performed at MS $k = 10$ for a part to be installed on GT $g = 2$. Then, at the two next MSs, the warehouse is still empty and two more purchase actions must be performed. The MRC solution

$$Q(\mathbf{S}_z, A_z) \longleftarrow Q(\mathbf{S}_z, A_z) + (\gamma\lambda)^{(k-z)}\alpha_n \cdot [-C_k + \gamma Q(\mathbf{S}_{k+1}, A_{k+1}) - Q(\mathbf{S}_k, A_k)] \forall z \in \{1, \ldots, k\} \quad (22)$$

Table 1.: Initial scenario and parameters

G	Z	W	R	C^{Scrap}	$C^{rep}(r=1)$	$C^{rep}(r=2)$	C^{pur}	γ
2	10	3	3	0	90	50	100	0.99

entails a final maintenance cost of 1290 and a discounted cost of about 1161, in arbitrary units: the cost contributions are due to 5 purchases of new parts, 5 repairs of parts with $r = 2$ remaining cycles and 6 repairs of parts with $r = 1$ remaining cycles.

The part flow solution given by the application of the MRC rule is compared with that provided by the ILP and RL optimal solution, which is summarized in Table 2 (the parameters used for the RL optimal solution have been set by a series of experiments; in particular, $\lambda = 0.8$). The application of the optimal policy found yields a final maintenance cost of 1150 and a discounted cost of 1033, in arbitrary units (last row of Table 3). These values are smaller than those found by the MRC policy (last row of Table 2). Thus, MRC is not an optimal policy.

To justify the difference, we can note that the optimal solution found by ILP and RL entails 7 purchase actions, 9 repair of parts with RUL $= 2$ and no repair of parts with RUL $= 1$. Then, although purchase actions are more expensive than repair actions of parts with RUL$= 1$, ($C^{pur} = 100 > C^{rep}(r = 1) = 90$), the former entail a larger final saving than the latter because they allow re-using the parts in different cycles. Notice also that the purchase action performed under the optimal policy at time $k = 17$ is less expensive than the repair action performed on a part with RUL$= 1$ at time $k = 4$ under the MRC policy (row 17 of Table 3, ($PV_{17} = 85.1$) vs row 4 of Table 2 ($PV_4 = 87.3$).

6. Discussion and Conclusions

In this work, we have framed a SDP for flow management of GT parts. Its application to a scaled-down case study derived from industrial practice has shown that MRC fails to give the most profitable part flow management policy, especially when the GTs are expected to work for a long time and the time value of money is non-negligible. In fact, the discount rate of future expenditures strongly affects the profitability of the policy, leading to very different optimal solutions even under the same initial warehouse and GTs' conditions.

Although the RL and ILP optimal policies are the same, there are fundamental differences between the two approaches. First, convergence of RL to the optimal solution is guaranteed only on an infinite number of simulations, whereas ILP framework always guarantees the optimality. However, ILP may not be used in industrial practice, due to the modeling effort it requires, which can entail significant model changes upon small difference in the optimization problem. Moreover, ILP is not applicable to the cases in which the complexity of the environment cannot be captured by linear constraints.

On the contrary, RL is a model-free method, which does not require the knowledge of the updating dynamics. This allows easily encoding additional features of the specific real applications, as it acts on the simulation of the decision process and, thus, selects actions from those feasible, only. For example, although here not considered, the complexity of the real industrial applications requires SDP to encode many additional GT operational aspects, such as the possibility of inspecting the parts without performing maintenance (i.e., condition-based maintenance), the different duration of the maintenance intervals for parts of different technologies, the constraints on the shareability of the parts on GTs with different operation temperatures, etc. Accounting for these GT operation features requires encoding constraints about the actions that can be taken in each state, which are really difficult to set in linear programming frameworks.

Moreover, RL algorithms allow encoding the aleatory uncertainties, e.g., in the failure times of the GT parts or in the non-negligible duration of the inspection cycle, more easily than the other algorithms.

The proposed RL framework suffers from some limitations that can still prevent its full application to the industrial practice in the current form: in complex problems, the state-space becomes very large, whereby the tabular representation of the state-action value function is not practicable. For this, action-value approximation techniques can be used, instead of the tabular approach hereby presented. This allows generalizing the state description, e.g., by removing the constraints on the maximum number of parts available in the warehouse for each RUL level or considering real-

Table 2.: MRL policy

k	$w_{1,k}$	$w_{2,k}$	$w_{3,k}$	RUL@GT$g=1$	RUL@GT$g=2$	RUL Installed Part	Repair	Purchase	C_k	PV_k
1	2	1	1	**2**	1	3	Y	N	50	50
2	2	2	0	3	**0**	2	N	N	0	0
3	2	1	0	**2**	2	2	Y	N	50	49.005
4	2	1	0	2	**1**	2	Y	N	90	87.327
5	3	0	0	**1**	2	1	Y	N	90	86.453
6	3	0	0	1	**1**	1	Y	N	90	85.589
7	3	0	0	**0**	1	1	N	N	0	0
8	2	0	0	1	**0**	1	N	N	0	0
9	1	0	0	**0**	1	1	N	N	0	0
10	0	0	0	1	**0**	3	N	Y	100	91.351
11	0	0	0	**0**	3	3	N	Y	100	90.438
12	0	0	0	3	**2**	3	Y	Y	150	134.300
13	0	1	0	**2**	3	2	Y	N	50	44.319
14	0	1	0	2	**2**	2	Y	N	50	43.876
15	0	1	0	**1**	2	2	Y	N	90	78.187
16	1	0	0	2	**1**	1	Y	N	90	77.405
17	1	0	0	**1**	1	1	Y	N	90	76.631
18	1	0	0	1	**0**	1	N	N	0	0
19	0	0	0	**0**	1	3	N	Y	100	83.451
20	0	0	0	3	**0**	3	N	Y	100	82.617
-	-	-	-	**2**	3	-	-	TOT	1350	1161

Table 3.: RL and ILP policies

k	$w_{1,k}$	$w_{2,k}$	$w_{3,k}$	RUL@GT$g=1$	RUL@GT$g=2$	RUL Installed Part	Repair	Purchase	C_k	PV_k
1	2	1	1	**2**	1	1	Y	N	50	50
2	1	2	1	1	**0**	1	N	N	0	0
3	0	2	1	**0**	1	2	N	N	0	0
4	0	1	1	2	**0**	3	N	N	0	0
5	0	1	0	**1**	3	2	N	N	0	0
6	0	0	0	2	**2**	3	Y	Y	150	142.64
7	0	1	0	**1**	3	2	N	N	0	0
8	0	0	0	2	**2**	3	Y	Y	150	139.810
9	0	1	0	**1**	3	2	N	N	0	0
10	0	0	0	2	**3**	3	Y	Y	150	137.02
11	0	1	0	**1**	3	2	N	N	0	0
12	0	0	0	2	**2**	3	Y	Y	150	134.300
13	0	1	0	**1**	3	2	N	N	0	0
14	0	0	0	2	**2**	3	Y	Y	150	131.682
15	0	1	0	**1**	3	2	N	N	0	0
16	0	0	0	2	**2**	3	Y	Y	150	129.008
17	0	1	0	**1**	3	3	N	Y	100	85.146
18	0	1	0	3	**2**	2	Y	N	50	42.147
19	0	1	0	**2**	2	2	Y	N	50	41.726
20	0	1	0	2	**1**	2	N	N	0	0
-	-	-	-	**1**	2	-	-	TOT	1150	1033

valued RUL estimations.

These results pave the way to research work to systemically address the part flow management issue.

References

Andersen, E. D. and K. D. Andersen (1995). Presolving in linear programming. *Mathematical Programming 71*(2), 221–245.

Bertsimas, D. and J. Tsitsiklis (1997). *Introduction to Linear Optimization* (1st ed.). Athena Scientific.

Boyce, M. (2012). *Gas Turbine Engineering Handbook*.

Cornuéjols, G. (2008). Valid inequalities for mixed integer linear programs. *Mathematical Programming 112*(1), 3–44.

Gollier, C. (2002). Time horizon and the discount rate. *Journal of Economic Theory 107*(2), 463–473.

Li, Z., Z. Ding, and M. Wang (2017, April). Optimal bidding and operation of a power plant with solvent-based carbon capture under a co 2 allowance market: A solution with a reinforce-

ment learning-based sarsa temporal-difference algorithm. *Engineering 3*(2), 257–265.
Nemhauser, G. and L. Wolsey (1999). Integer programming and combinatorial optimization. In *Wiley, 1988*. Springer.
Schrijver, A. (1986). *Theory of Linear and Integer Programming*. New York, NY, USA: John Wiley & Sons, Inc.
Sutton, R. S. and A. G. Barto (2018). *Reinforcement Learning: An Introduction* (Second ed.). The MIT Press.
Szepesvári, C. (2010). Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning 4*(1), 1–103.
Wang, Y.-H., T.-H. S. Li, and C.-J. Lin (2013, October). Backward q-learning: The combination of sarsa algorithm and q-learning. *Eng. Appl. Artif. Intell. 26*(9), 2184–2193.
Whitehead, S. D. and L.-J. Lin (1995, February). Reinforcement learning of non-markov decision processes. *Artif. Intell. 73*(1-2), 271–306.