Component-Wise Markov Decision Process for Solving Condition Based Maintenance of Large Multi-Component Systems with Economic Dependence

Vipul Bansal [a], Yong Chen [b,], and Shiyu Zhou [a]
[a] Department of Industrial and Systems Engineering,
University of Wisconsin, Madison, WI53706
[b] Department of Industrial and Systems Engineering,
University of Iowa, Iowa City, IA 52242

Abstract

Condition-based maintenance of multi-component systems is a prevalent engineering problem due to its effectiveness in reducing the operational and maintenance costs of the system. However, developing the exact optimal maintenance decisions for the large multi-component system is computationally challenging, even not feasible, due to the exponential growth in system state and action space size with the number of components in the system. To address the scalability issue in CBM of large multi-component systems, we propose a Component-Wise Markov Decision Process(CW-MDP) and an Adjusted Component-Wise Markov Decision Process (ACW-MDP) to obtain an approximation of the optimal system-level CBM decision policy for large systems with heterogeneous components. We propose using an extended single-component action space to model the impact of system-level setup cost on a component-level solution. The theoretical gap between the proposed approach and system-level optima is also derived. Additionally, theoretical convergence and the relationship between ACW-MDP and CW-MDP are derived. The study further shows extensive numerical studies to demonstrate the effectiveness of component-wise solutions for solving large multi-component systems.

Keywords: Condition-Based Maintenance, Multi-Component System, Markov Decision Process

1 Introduction

In practice, production/service systems often consist of multiple components, for example, various machines in a production system, a fleet of trucks in a transportation system, or wind turbines in a windmill farm. In this work, we focus on the condition-based maintenance problem for a multi-component system that can potentially contain a large number of

components. Maintenance decision for such a system is a challenging decision-making problem. The complex nature of the problem arises from the dependence of the decisions at a single component level due to stochastic, economic, or structural dependencies among components. It is well known that, in most cases, the optimal maintenance decision at the single component level is not optimal for the whole system. Unfortunately, developing the exact optimal maintenance decisions for the whole system is computationally challenging, even not feasible, due to the exponential growth in system state and action space size with the number of components in the system.

Maintenance decision-making of multi-component systems is an active research area with a large amount of available literature. Some excellent surveys of this field can be found in (Keizer et al., 2017a; Wang and Chen, 2016; Dehghani Ghobadi et al., 2022; Pinciroli et al., 2023). In this work, we will focus on the condition-based maintenance (CBM) method for multi-component systems. In CBM, the components' health is monitored using sensory data collected in real time, and maintenance decisions are made based on the machine's health condition. Intuitively, CBM will outperform time/age-based maintenance in most cases because more real-time information has been taken into consideration in the decision-making process (Koochaki et al., 2013).

For CBM decision-making, we first need to select a degradation model to describe the underlying health deterioration path followed by a component in the system. A popular choice of degradation model in maintenance decision-making is the discrete Markov process, in which the health condition of the component is represented by a few discrete states, and the evolving of the health condition is represented by the transition between states (Alaswad and Xiang, 2017). There are some advantages of using a discrete Markov process as the degradation model. First, the discrete Markov process is quite a flexible model. In fact, other degradation models, such as the continuous degradation path models, can be easily approximated by the discrete Markov processes as shown in (Bouvard et al., 2011; Peng et al., 2021; Barbera et al., 1999). A recent work showed that a discrete Markov process effectively approximates continuous degradation processes using only a few states (Andersen, 2022). Second, with the discrete-state Markov process representation, we can naturally formulate the maintenance decision-making problem into a Markov Decision Process(MDP) and obtain the optimal maintenance policy under the MDP framework. Indeed, many

research works (Liu et al., 2013; Xu et al., 2022; Zhao et al., 2019) have been reported to solve the optimal CBM problem for a single component using MDP.

Limited work is available on using MDP to solve CBM problem for multi-component systems. The reason is that the state space and action space of a multi-component system is extremely large, which makes it computationally expensive to obtain the exact MDP solution. For instance, a system consisting of M components with L degradation states for each component, and each component having only two actions: repair or no repair, would lead to a state space of size L^M and an action space of size 2^M. Solving an MDP with such a state and action space is practically infeasible for systems with a large number of components. Thus, most existing MDP-based CBM methods for a multi-component system can only work for a system with a very small number (e.g., two) of components (Keizer et al., 2017b; Zhang et al., 2011). A recent study showed that it is only feasible to obtain the exact MDP solution for up to 7 components and 12 degradation levels (Andersen et al., 2022). To address the scalability issue in CBM of large multi-component systems, several approaches have been proposed. One approach is to use some heuristic policies to provide a sub-optimal solution to the problem. For example, a popular heuristic rule is the (n, N)-policy (Andersen, 2022) for a system with identical components. This rule specifies that if one or more component is at or beyond health degradation level N, then all the components at or beyond health degradation level n will be repaired. Obviously, this rule can be applied to a system with a very large number of components. However, such a rule cannot be applied to a system with heterogeneous components, where "heterogeneous" means that the state transition dynamics of a component is different from that of another component. In addition, the conditions under which the performance of this rule is good are not very clear. There are other threshold-based heuristic policies for CBM (Xia et al., 2013; Zhang and Zeng, 2015). Similarly, these heuristic policies are sub-optimal and cannot be applied to a system with heterogeneous components. In recent years, the deep reinforcement learning (DRL) method has been used to obtain the optimal maintenance policy for multi-component system (Zhang and Si, 2020; Yousefi et al., 2022). DRL is a black-box approximation method that can solve the CBM problem for a relatively large multi-component system when compared with the systems that exact MDP solvers can handle. However, this approach still faces significant computational challenges due to the curse of dimensionality because

of exponentially increasing state and action space. A larger state space necessitates a significant number of simulation trials to learn a good function approximation effectively (Wong et al. (2023)). Thus, DRL cannot be applied to very large multi-component systems. One way to reduce the computational space is by reducing state space by considering groups of identical components in the system and reducing action space by using constraints in the system. Using these ideas, (Liu et al., 2020) solved for an optimal policy using actor-critic for 14 components by considering 5 groups of identical components to reduce the state space and by reducing action space to 8 actions by using resource constraints. Still, these approximations cannot help solve systems with a very large number of components and cannot be applied to a general system with heterogeneous components. Another approach of addressing the scalability issue is to use online planning methods such as the Monte Carlo Tree Search (MCTS) to obtain a good action only for the current system state instead of obtaining the complete action policy for all the possible system states. MCTS has been used in maintenance decision-making problems recently(Barata et al., 2002; Marseguerra et al., 2002). Similar to DRL, the online planning approach cannot yet handle very large systems. If we want to obtain high-quality solutions, the computational load is prohibitively intensive for large systems.

In this work, we propose a scalable approach to solve the CBM decision-making problem for systems with a large number of heterogeneous components with economic dependence. The component degradation is modeled by discrete Markov processes. To capture the economic dependence among the components, we propose a decomposition of the system reward function and an extension of the action space to reflect the new reward function. Then, we propose a two-step decision-making scheme to find an approximation of the optimal system-level CBM decision policy. In the first step, we obtain the component level Q functions under the extended action space and the decomposed reward function by solving the component level MDP. In the second step, we will select the best system-level actions based on the component level Q functions obtained in the first step by explicitly taking into account the economic dependence. This proposed approach is a heuristic approximation approach. However, different from existing heuristic approaches, our proposed approach is based on the component-level MDP solution under the extended action space. Thus, this approach can address heterogeneous components naturally. In addition, the theoretical gap

between the solution obtained by our proposed approach and the exact optimal solution can be obtained. The computational time of the proposed algorithm grows linearly with the number of components in the system, allowing us to scale for extremely large-scale problems.

The paper is organized as follows: In Section 2, the mathematical formulation is provided, in which the maintenance decision-making for a multi-component system is formulated as an MDP. In Section 3, we propose a scalable solution to the multi-component CBM problem based on component-wise MDP. In Section 4, the theoretical properties of the proposed approach are investigated. Section 5 presents a comprehensive numerical study to illustrate the advantage of the proposed method. Section 6 concludes the paper and provides some discussion of future works.

2 Problem Formulation

We consider a system consisting of M heterogeneous components, i.e., different components follow different transition dynamics. The system is inspected at equal time intervals. The inspection cost is ignored. At each inspection, we observe the degradation state of each component in the system. The degradation state s_i of a component i belongs to set $\mathcal{S}_c^{(i)} = \{1, 2, 3, ..., L_i\}$, where state 1 corresponds to the new state of the component and L_i corresponds to the failure state of the component.

After each inspection, we take two possible actions for each component $\{0, 1\}$, where 0 corresponds to retaining the component and 1 stands for replacement of the component. After replacement, a component returns to state 1. At degradation state L_i, we always take action $a_i = 1$, which means a failed component is immediately replaced with a new component with a corrective replacement cost C_b. If a component is replaced at a state $s_i < L_i$, it is replaced at a preventive replacement cost $C_m^{(i)}$ such that $C_m < C_b$. Thus, we can define the cost of an individual component at state s_i and action a_i as

$$r_i(s_i, a_i) = \begin{cases} -C_m & s_i \neq L, a_i = 1 \\ -C_b & s_i = L \\ 0 & \text{otherwise} \end{cases} . \tag{1}$$

We would like to point out that in (1), we assume each component has the same C_m value

and the same C_b value, i.e., $C_m^{(i)} = C_m$ and $C_b^{(i)} = C_b$, while $C_m < C_b$. This assumption is just for notational convenience. The proposed method can be easily extended to cases where the repair and replacement costs are different for different components. The degradation of each component follows a discrete-time Markov decision process with transition probability from a state s_i to s_i' be $T_i(s_i'|s_i, a_i = 0)$. Additionally, the transition probability satisfies $T_i(s_i'|s_i, a_i = 0) = 0$ for $s_i' \leq s_i$, and $\sum_{s_i'} T_i(s_i'|s_i, a = 0) = 1$. For any replacement, i.e. $s_i = L_i$ or $a_i = 1$ we let $T(s_i'|s_i, a_i = 1) = T(s_i'|s_i = L_i, a_i = 1) = T(s_i'|s_i = 1, a_i = 0)$, since we assume that all replacements are instantaneous and occurs at the beginning of a time period. Thus, the replaced component returns to the state 1 and then performs a transition for the period from state 1.

Using state s_i and action a_i for the individual component, we can form a state vector $\mathbf{s}$ as $< s_1, s_2, s_3, ..., s_M >$, and action vector $\mathbf{a}$ as $< a_1, a_2, ...a_M >$. We can denote the complete system-level state and action space as $\mathcal{S}$ and $\mathcal{A}$, respectively, such that $\mathbf{s} \in \mathcal{S}$ and $\mathbf{a} \in \mathcal{A}$. The system-level transition probability of transiting from a state $\mathbf{s}$ and action $\mathbf{a}$ to state $\mathbf{s}'$ can be formulated using *transition independence* of components as:

$$T(\mathbf{s}'|\mathbf{s}, \mathbf{a}) = \prod_{i=1}^{M} T_i(s_i'|s_i, a_i). \tag{2}$$

If there is any replacement in the system, there is an additional setup cost of C_s. This setup cost induces an *economic dependence* between components, encouraging components to get replaced simultaneously. We can define the system-level reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as:

$$R(\mathbf{s}, \mathbf{a}) = \begin{cases} -C_s + \sum_{i=1}^{M} r_i(s_i, a_i) & \text{Any Replacement} \\ 0 & \text{No Replacement} \end{cases} \tag{3}$$

Using the state space, action space, reward function and transition dynamics given above, we formulate the condition-based maintenance problem as a Markov Decision Process (MDP)$\{\mathcal{S}, \mathcal{A}, R, T, \gamma\}$, where $\gamma \in (0, 1]$ is the discount factor. Starting at a given state $\mathbf{s}_0$ at time t=0, we find a policy π from a set of all policies Π such that it maximizes the expected total discounted rewards for the system $V^*(s_0)$ as

$$V^*(s_0) = \max_{\pi \in \Pi} E\left[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{s}_t, \pi(\mathbf{s}_t)) \right] \ \forall s_0 \in \mathcal{S}. \tag{4}$$

The optimal Q-function Q^* and value function V^* can be obtained by value iteration(Puterman, 2014) over the entire state and action space such that:

$$Q^*(\mathbf{s}, \mathbf{a}) = R(\mathbf{s}, \mathbf{a}) + \gamma \mathbf{E}_{\mathbf{s}' \sim T(\mathbf{s}'|\mathbf{s},\mathbf{a})}[V^*(\mathbf{s}')] \tag{5}$$

$$V^*(\mathbf{s}) = \max_a Q^*(\mathbf{s}, \mathbf{a}). \tag{6}$$

Using the optimal Q^* we can obtain an optimal policy $\pi^*(\mathbf{s})$ for a given state $\mathbf{s}$ as:

$$\pi^*(\mathbf{s}) = \arg\max_{\mathbf{a}} Q^*(\mathbf{s}, \mathbf{a}). \tag{7}$$

Although value iteration guarantees an exact optimal solution to the problem, the scalability of this approach is poor. The above formulation leads to a prohibitively large state and action space, i.e., $|\mathcal{S}|$ and $|\mathcal{A}|$ would be $\prod_{i=1}^{M} L_i$ and 2^M, respectively. When using value iteration, iteration over all states and actions is required; this makes obtaining a solution challenging, even for a small-scale system. In the next section, we propose a scalable solution method for this problem.

3 Scalable Component-Wise Solution for CBM of Multi-Component Systems

To address the scalability issue for CBM of multi-component systems, we could find an optimal CBM decision for each component, called the "component-wise" solution, and then combine the component-wise decision to form the decision for the whole system. However, the replacement of components has an economic dependence due to setup cost C_s. Clearly, if we simply ignore the setup cost in the component-wise decision (i.e., find the optimal CBM decision for each component under the cost function given in (1)), then the component-wise solution will not lead to a good solution for the system level CBM problem. To deal with this issue, we can develop a component-wise solution by considering the impact of a single component's decision on the entire system. In other words, we could bring the setup cost into the component-wise CBM problem so that the economic dependence among components is taken into consideration to a certain extent when we make a component-wise decision.

Following the above intuition, we propose to reformulate the cost function as in (8) and establish an *extended single-component action space* as $\mathcal{A}^e = \{a_{0,0}, a_{0,1}, a_{1,1}\}$ for the

reformulated cost function,

$$R_I(s_i, \alpha_i) = \begin{cases} -C_b - \frac{C_s}{M} & \text{if } s_i = L_i \\ 0 & \text{if } \alpha_i = a_{0,0} \ \& \ s_i \neq L_i \\ -\frac{C_s}{M} & \text{if } \alpha_i = a_{0,1} \ \& \ s_i \neq L_i \\ -C_m - \frac{C_s}{M} & \text{if } \alpha_i = a_{1,1} \ \& \ s_i \neq L_i \end{cases}. \tag{8}$$

In the above equation, $\alpha_i \in \mathcal{A}^e$ is the component-wise action for component i, $a_{0,0}$ corresponds to the action that a component is not getting replaced and there is no setup cost, whereas, $a_{1,1}$ and $a_{0,1}$ correspond to the action that a component is replaced and not replaced, respectively, and there is setup cost. This reward function states that if there is one or more replacements in the system, the setup cost is distributed evenly to all components in the system. Distributing setup cost evenly among components is just a *mathematical* way to adjust the component-level cost so that total system-level cost will be correctly obtained by summing all component-level costs. The transition dynamics for actions in extended single-component action space are defined as: $T(s'|s, a_{0,0}) = T(s'|s, a_{0,1}) = T(s'|s, a_i = 0)$ and $T(s'|s, a_{1,1}) = T(s'|s, a_i = 1)$. Because the extended single-component actions do not change the state transition dynamics, the following result can be obtained.

Result 1. *For every system state* $\boldsymbol{s}$ *and action* $\boldsymbol{a}$*, there exists a valid combination of extended single-agent actions* $< \alpha_1, \alpha_2, \alpha_3 ....., \alpha_M >$ *such that:*

$$\sum_{i=1}^{M} R_I(s_i, \alpha_i) = R(\boldsymbol{s}, \boldsymbol{a}), \tag{9}$$

$$T(\boldsymbol{s'}|\boldsymbol{s}, \boldsymbol{a}) = \prod_{i=1}^{M} T_i(s_i'|s_i, \alpha_i). \tag{10}$$

The above result is obvious: if there is no replacement in the system, we can take $\alpha_i = a_{0,0}$ for all $i = 1, ..., M$, while if k, $k > 0$, components are replaced in the system, we can take action $a_{1,1}$ for the k replaced components and the action $a_{0,1}$ for the rest of the components.

With the new reward function and the extended action space, we can develop an approximated component-wise solution to the CBM of multi-component systems.

3.1 CW-MDP: Component-Wise Markov Decision Process

For the state space $\mathcal{S}_c^{(i)}$, action Space $\mathcal{A}^e$, and reward structure R_I we can formulate a component-wise MDP $\{\mathcal{S}_c^{(i)}, \mathcal{A}^e, R_I, T_i, \gamma\}$. We can develop a policy by maximizing total discounted rewards for each component in the system using this MDP. In this component-wise MDP, the setup cost is considered in the reward function, so we expect the component-wise MDP solution will provide a better approximation to the system-level MDP problem. That being said, please note that CW-MDP is just an approximation and cannot provide the exact solution to the system-level CBM problem.

We can solve the CW-MDP problem to compute component-wise Q-function Q^c as

$$Q^c(s_i, \alpha_i) = R_I(s_i, \alpha_i) + \gamma \mathbf{E}_{s_i' \sim T_i(s_i'|s_i, \alpha_i)}[V^c(s_i)] \tag{11}$$

where $V^c(s_i)$ is the component-wise value function defined as:

$$V^c(s_i) = \max_{\alpha_i \in \mathcal{A}^e} Q^c(s_i, \alpha_i) \tag{12}$$

The above Q^c and V^c can be computed for each component of the system using their respective transition matrices T_i. For CW-MDP, the state space only consists of the health states of a single component; thus, it is quite small. We use the standard value iteration approach to solve it (Puterman, 2014). For the sake of convenience, we list the value iteration method in Algorithm 1. In Algorithm 1, $Q^c_{(i)}$ and $V^c_{(i)}$ represent the Q^c and V^c-function of the i^{th} component of the system, respectively. For the Q^c function, we have the following result:

Result 2. *For any state s_i given $s_i \neq L$, we have:*

$$Q^c(s_i, a_{0,0}) - Q^c(s_i, a_{0,1}) = \frac{C_s}{M}. \tag{13}$$

This result is obvious if we consider the fact that the state transition under $a_{0,0}$ or $a_{0,1}$ is the same, and $R_I(s_i, a_{0,0}) - R_I(s_i, a_{0,1}) = \frac{C_s}{M}$. This result indicates that $Q^c(s_i, a_{0,0}) > Q^c(s_i, a_{0,1})$, which means that to maximize the rewards of an individual component, we will always choose action $a_{0,0}$ over action $a_{0,1}$. In other words, we will never choose action $a_{0,1}$ as an optimal action in CW-MDP. However, it is obvious that in the optimal system-level solution, $a_{0,1}$ should be selected for the retained component when there is at least one replacement in the system. This implies we need a suitable optimal action selection method for the whole system using the component-wise Q^c.

Algorithm 1 Component-Wise Markov Decision Processes

procedure CWMDP($\{T_1, T_2, ...T_M\}, \gamma$)
  $V^c_{(1)},\ V^c_{(2)}, ...,\ V^c_{(M)}$: Randomly initialized
  for all $i \in \{1, 2, ..., M\}$ **do**
    Repeat
    for all $s_i \in \mathcal{S}_c^{(i)}$ **do**
      for all $\alpha_i \in \mathcal{A}^e$ **do**
        $Q^c_{(i)}(s_i, \alpha_i) \leftarrow R_I(s_i, \alpha_i) + \sum_{s_i' \in \mathcal{S}} T_i(s_i'|s_i, \alpha_i)[V^c_{(i)}(s_i')]$
      end for
      $V^c_{(i)}(s_i) \leftarrow \max_{\alpha_i} Q^c_{(i)}(s_i, \alpha_i)$
    end for
    Iterate till $V^c_{(i)}$ converges
  end for
  return $Q^c_{(1)},\ Q^c_{(2)}, ...,\ Q^c_{(M)}$
end procedure

3.2 System-Level Action Selection using CW-MDP

Instead of selecting the actions that maximize each individual Q^c, which will fall short of providing the optimal actions at the system level, we can try to find an action vector $\mathbf{a}$ for a given state $\mathbf{s}$ that maximizes the sum of Q^c of each individual component. The action vector $\mathbf{a}$ can be mapped to a valid combination of extended single agent actions α_i using Result 1. We propose the following way of selecting actions to generate a policy $\pi_{cw}(\mathbf{s})$ as

$$\pi_{cw}(\mathbf{s}) = \begin{cases} \arg\max_{\alpha_i \in \overline{\mathcal{A}}^e} Q^c_{(i)}(s_i, \alpha_i) \forall i & \text{if } \sum_i \max_{\alpha_i \in \overline{\mathcal{A}}^e} Q^c_{(i)}(s_i, \alpha_i) \geq \sum_i Q^c_{(i)}(s_i, a_{0,0}) \\ & \quad \text{or } s_i = L_i \\ a_{0,0}\ \forall i & \text{otherwise} \end{cases} \tag{14}$$

where $\overline{\mathcal{A}}^e = \{a_{0,1}, a_{1,1}\}$.The intuition behind (14) is to explicitly compare the value of $\sum_i Q^c_{(i)}$ between two scenarios: no replacement in the system or at least one replacement in the system. One subtle point we want to mention is that if $\sum_i \max_{\alpha_i \in \overline{\mathcal{A}}^e} Q^c_{(i)}(s_i, \alpha_i) \geq \sum_i Q^c_{(i)}(s_i, a_{0,0})$ holds, then action $a_{1,1}$ is selected for at least one component. This is due to the result 2, which indicates that if no $a_{1,1}$ is selected, then $\sum_i \max_{\alpha_i \in \overline{\mathcal{A}}^e} Q^c_{(i)}(s_i, \alpha_i) \geq \sum_i Q^c_{(i)}(s_i, a_{0,0})$

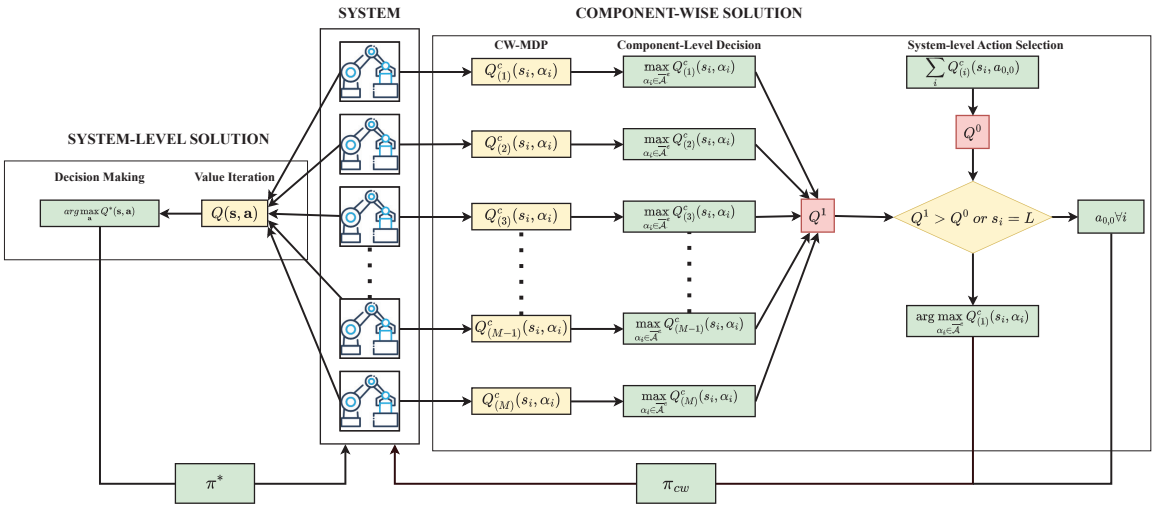

Figure 1: Schematic Representation of CW-MDP with system-level action selection

will not hold. Thus, $< \alpha_1, \alpha_2, \alpha_3 ....., \alpha_M >$ selected in the first equation in (14) always gives a valid combination of extended single-agent actions.

Algorithm 2 gives implementation details for the system-level action selection procedure. The initial step involves checking whether any component is in a failure state $s_i = L_i$. In this scenario, the presence of a failure implies the need for maintenance, consequently incurring a setup cost. As a result, all components are limited to choosing actions $a_{0,1}$ and $a_{1,1}$, reflecting the necessary maintenance actions under such conditions. Conversely, if corrective maintenance is not required, we proceed to calculate Q^0 and Q^1. The subsequent decision-making process involves comparing Q^0 and Q^1. If $Q^0 > Q^1$, then the system-wide action chosen is $a_{0,0}$ for all components. Alternatively, if Q^1 is greater than Q^0, the system-level action becomes $arg\max_{\alpha_i \in \overline{\mathcal{A}}^e} Q^c_{(i)}(s_i, \alpha_i)$ for each component. This approach simplifies the complex combinatorial optimization problem by ensuring a valid combination of the extended single-agent action space is selected at the system level. A diagram of the overview of the CW-MDP-based solution is also provided in Figure 1.

In Algorithm 1, the estimation of V^c in CW-MDP involves maximizing over actions $\{a_{0,0}, a_{0,1}, a_{1,1}\}$, but CW-MDP never chooses action $a_{0,1}$ in the maximization step. However, in the system-level action selection process, action $a_{0,1}$ can be chosen for components that are not being replaced, given there is maintenance in the system. It is critical to note that system-level action selection allows us to make system-level decisions from component-wise

Q^c estimates with just M separate binary decisions. This makes the solution feasible for large-scale problems.

Algorithm 2 System-level Action Selection Algorithm

procedure DECISION($\{Q^c_{(1)},\ Q^c_{(2)}, ...,\ Q^c_{(M)}\}, \mathbf{s}$)
  if $\exists s_i = L_i$ **then**
    for all $i \in \{1, 2, ..., M\}$ **do**
      $\alpha_i \leftarrow \arg\max_{\alpha_i \in \{a_{0,1}, a_{1,1}\}} Q^c_{(i)}(s_i, \alpha_i)$ ▷ Setup cost exists in system
    end for
    $\alpha_i \mapsto a_i$
    return $< a_1, a_2, ...a_M >$
  end if
  $Q^0 \leftarrow \sum_i Q^c_{(i)}(s_i, a_{0,0})$
  $Q^1 \leftarrow 0$
  for all $i \in \{1, 2, ..., M\}$ **do**
    $\alpha_i \leftarrow \arg\max_{\alpha_i \in \{a_{0,1}, a_{1,1}\}} Q^c_{(i)}(s_i, \alpha_i)$ ▷ Select best action for individual machine
    $Q^1 \leftarrow Q^1 + Q^c_{(i)}(s_i, \alpha_i)$
  end for
  if $Q^0 > Q^1$ **then**
    $\alpha_i \leftarrow a_{0,0} \forall i$
    $\alpha_i \mapsto a_i$
    return $< a_1, a_2, ...a_M >$
  else
    $\alpha_i \mapsto a_i$
    return $< a_1, a_2, ...a_M >$
  end if
end procedure

3.3 ACW-MDP: Adjusted Component-Wise Markov Decision Processes

One weakness of the method presented in Section 3.1 is to imitate the system behavior at

the component level $Q^c_{(i)}$ estimates. Due to Result 2 and the max operation in (12), in solving CW-MDP, we will avoid selecting the action $a_{0,1}$ when we estimate $V^c(s_i)$. This will impact the solution quality even if we use a system-level action selection remedy as described in Section 3.2.

To address this issue, we propose an Adjusted Component-Wise Markov Decision Process (ACW-MDP) to consider all the possible actions in the extended single-component action space when we solve for component-wise MDP. In ACW-MDP, we adjust Q^c and V^c, such that instead of maximization over the extended component-wise action space, we use the probability by which a component i chooses an action α_i at a given state s_i. For this, we define an adjusted component-wise Q-function $Q^{c'}$ as:

$$Q^{c'}(s_i, \alpha_i) = R_I(s_i, \alpha_i) + \gamma \mathbf{E}_{s'_i \sim T(s'_i|s_i,\alpha_i)}[V^{c'}(s'_i)], \tag{15}$$

where $V^{c'}$ is an adjusted component-wise value function defined as:

$$V^{c'}(s_i) = \mathbb{P}(a_{0,0}|s_i)Q^{c'}(s_i, a_{0,0}) + \mathbb{P}(a_{0,1}|s_i)Q^{c'}(s_i, a_{0,1}) + \mathbb{P}(a_{1,1}|s_i)Q^{c'}(s_i, a_{1,1}), \tag{16}$$

where $\mathbb{P}(a_{0,0}|s_i)$, $\mathbb{P}(a_{0,1}|s_i)$, and $\mathbb{P}(a_{1,1}|s_i)$ represent the probability of choosing actions $a_{0,0}$. $a_{0,1}$ and $a_{1,1}$ for component i given state s_i.

Understanding that $V^{c'}$ function differs from V^c function is critical. The V^c function maximizes a component's expected total discounted reward. In contrast, the $V^{c'}$ function gives the expected total discounted reward for a component since a component chooses all actions with some probability instead of choosing one action at a time.

To compute $Q^{c'}(s_i, \alpha_i)$, estimation of probabilities $\mathbb{P}(a_{0,0}|s_i)$, $\mathbb{P}(a_{0,1}|s_i)$, and $\mathbb{P}(a_{1,1}|s_i)$ is required. This can be challenging since probability $\mathbb{P}(\alpha_i|s_i)$ is impacted by the state and actions of all other components in the system. Accounting for the states of other components in estimating $\mathbb{P}(\alpha_i|s_i)$ would lead to a full system wide solution instead of a *component-wise* solution. However, we know that the full system wide solution will encounter the scalability issue, which is actually the challenge we try to address in this work. Consequently, we ignore the states of all other components while estimating the probability $\mathbb{P}(\alpha_i|s_i)$. Intuitively, we can view the probability $\mathbb{P}(\alpha_i|s_i)$ as an approximation of the probability $\mathbb{P}(\alpha_i|s_i$, state of other components) regardless the states of other components. Here, we propose estimating the probability $\mathbb{P}(\alpha_i|s_i)$ using softmax response(Kochenderfer

et al., 2022) as:

$$\mathbb{P}(\alpha_i|s_i) \propto e^{\lambda Q^{c'}(s_i,\alpha_i)} \tag{17}$$

Hence we can write:

$$\mathbb{P}(\alpha_i|s_i) = \frac{e^{\lambda Q^{c'}(s_i,\alpha_i)}}{\sum_{\alpha_i \in \{a_{0,0}, a_{0,1}, a_{1,1}\}} e^{\lambda Q^{c'}(s_i,\alpha_i)}} \tag{18}$$

We can further rewrite the above probability using additional $\min_{\alpha_i} Q^{c'}(s_i, \alpha_i)$ as:

$$\mathbb{P}(\alpha_i|s_i) = \frac{e^{\lambda (Q^{c'}(s_i,\alpha_i) - \min_{\alpha_i} Q^{c'}(s_i,\alpha_i))}}{\sum_{\alpha_i \in \{a_{0,0}, a_{0,1}, a_{1,1}\}} e^{\lambda (Q^{c'}(s_i,\alpha_i) - \min_{\alpha_i} Q^{c'}(s_i,\alpha_i))}} \tag{19}$$

This will not impact on the value of probability $\mathbb{P}(\alpha_i|s_i)$ but allows us to use a large value of λ without causing overflow in the numerical computation.

The parameter λ allows us to estimate the probability of choosing an action for a component. The above approximation allows us to use a single λ to decide the probability $\mathbb{P}(\alpha_i|s_i)$ for all state and action pairs of a component. For a given λ, we can use Algorithm 3 to get an estimate of the $Q^{c'}$ function.

In the above algorithm $Q^{c'}_{(i)}$ and $V^{c'}_{(i)}$ are the adjusted component-wise Q-function and value function of the i^{th} component in the system. Once we compute the $Q^{c'}$ function for each component in the system, we can also use system-level action selection by using Algorithm 2 and replacing Q^c with $Q^{c'}$. This adjusted policy π'_{cw} can be expressed as:

$$\pi'_{cw}(\mathbf{s}) = \begin{cases} \arg\max_{\alpha_i \in \overline{\mathcal{A}}^e} Q^{c'}_{(i)}(s_i, \alpha_i) \forall i & \text{if } \sum_i \max_{\alpha_i \in \overline{\mathcal{A}}^e} Q^{c'}_{(i)}(s_i, \alpha_i) \geq \sum_i Q^{c'}_{(i)}(s_i, a_{0,0}) \\ & \quad \text{or } s_i = L \\ a_{0,0} \ \forall i & \text{otherwise.} \end{cases} \tag{20}$$

It is critical to note that under the condition $s_i = L_i$, a component undergoes corrective maintenance and is replaced irrespective of the action taken. Due to the current formulation of the reward function, when considering a failed unit ($s_i = L_i$), the same failure reward is assigned to all actions, and system-level action selection will always replace a failed unit. For the tuning parameter λ, we can find an optimal λ^* that gives the best system-level discounted rewards by doing a search over a possible set of values of λ. This implies that we iteratively evaluate a generated policy $\pi'_{cw}(\mathbf{s})$ for a given λ to get system-level rewards and choose a λ^* that best fits the dynamics of a given system. We can mathematically define λ^*

Algorithm 3 Adjusted Component-Wise Markov Decision Processes

procedure ACW-MDP($\{T_1, T_2, ...T_M\}, \gamma$)
  for all $i \in 1, 2, ..., M$ **do**
    Repeat
    for all $s_i \in \mathcal{S}_c^{(i)}$ **do**
      for all $\alpha_i \in \{a_{0,0}, a_{0,1}, a_{1,1}\}$ **do**
        $Q_{(i)}^{c'}(s_i, \alpha_i) \leftarrow R_I(s_i, \alpha_i) + \sum_{s_i' \in \mathcal{S}} T_i(s_i'|s_i, \alpha_i)[V_{(i)}^{c'}(s_i')]$
      end for
      for all $\alpha_i \in \{a_{0,0}, a_{0,1}, a_{1,1}\}$ **do**
        $\mathbb{P}_{(i)}(\alpha_i|s_i) \leftarrow \frac{e^{\lambda(Q_{(i)}^{c'}(s_i,\alpha_i) - \min_{\alpha_i} Q_{(i)}^{c'}(s_i,\alpha_i))}}{\sum_{\alpha_i \in \{a_{0,0}, a_{0,1}, a_{1,1}\}} e^{\lambda(Q_{(i)}^{c'}(s_i,\alpha_i) - \min_{\alpha_i} Q_{(i)}^{c'}(s_i,\alpha_i))}}$
      end for
      $V_{(i)}^{c'}(s_i) \leftarrow \sum_{\alpha_i} \mathbb{P}_{(i)}(\alpha_i|s_i) Q_{(i)}^{c'}(s_i, \alpha_i)$
    end for
    Iterate till $V_{(i)}^{c'}$ converges
  end for
  return $Q_{(1)}^{c'},\ Q_{(2)}^{c'}, ...,\ Q_{(M)}^{c'}$
end procedure

as:

$$\lambda^* = \arg\max_{\lambda \in \mathcal{L}} E\left[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{s}_t, \pi'_{cw,\lambda}(\mathbf{s}_t))\right], \tag{21}$$

where $\mathcal{L}$ corresponds to a set of possible values of λ over which we perform a grid search to find the optimal λ^*, and $\pi'_{cw,\lambda}$ is the adjusted policy for a given λ.

4 Convergence Properties of CW-MDP and ACW-MDP

In this section, we will analyze a few properties of CW-MDP and ACW-MDP, specifically the convergence of the two methods and the relationship of the Q^c and $Q^{c'}$ functions to optimal system-level Q function. For ease of notation, we will use Q^c and V^c instead for $Q_{(i)}^c$ and $V_{(i)}^c$. Similarly, we will use $Q^{c'}$ and $V^{c'}$ instead of $Q_{(i)}^{c'}$ and $V_{(i)}^{c'}$. With a slight abuse of notation, let $\boldsymbol{V_t^c} \in \mathbb{R}^{|\mathcal{S}_c|}$ correspond to the V^c-function at the given t^{th} iteration step. And, $Q_t^c(s_i, .)$ corresponds to the Q^c-function estimated using $V_t^c(s_i)$ at a given t^{th} iteration step.

We can further express iterations for CW-MDP given in Algorithm 1 for a single component using $\mathcal{T}_m$ operator which iterates such that $\mathcal{T}_m \boldsymbol{V_t^c} = \boldsymbol{V_{t+1}^c}$, which is defined as:

$$(\mathcal{T}_m \boldsymbol{V_t^c})(s_i) = V_{t+1}^c(s_i) = \max_{\alpha_i}(Q_t^c(s_i, \alpha_i)) = \max_{\alpha_i} \Big(R_I(s_i, \alpha_i) + \gamma \sum_{s_i'} T(s_i'|s_i, \alpha_i) V_t^c(s_i) \Big) \tag{22}$$

For a *max* operator, we know that it satisfies the non-expansive property (Littman and Szepesvári, 1996) as:

$$|\max(Q_1^c(s_i, .)) - \max(Q_2^c(s_i, .))| \leq ||Q_1^c(s_i, .) - Q_2^c(s_i, .)||_\infty \tag{23}$$

The non-expansive property guarantees the convergence of operator $\mathcal{T}_m$ to a fixed unique solution $\boldsymbol{V^{c*}}$ such that $\mathcal{T}_m \boldsymbol{V^{c*}} = \boldsymbol{V^{c*}}$ and corresponding to this $V^{c*}(s_i)$ we will have a unique $Q^{c*}(s_i, .)$. The relationship between $Q^{c*}(s_i, \alpha_i)$ of all components in the system and the system level optimal Q-function $Q^*(\mathbf{s}, \mathbf{a})$ is given in the following proposition.

Proposition 1. *For a valid combination of extended single-agent actions* $< \alpha_1, \alpha_2, ..., \alpha_M >$ *corresponding to a system-level action* $\boldsymbol{a}$*, the error bound between* $Q^{c*}(s_i, \alpha_i)$ *and the optimal solution* $Q^*(\boldsymbol{s}, \boldsymbol{a})$ *is given by:*

$$\max_{\boldsymbol{s}} \max_{\boldsymbol{a}} |\sum_{i=1}^{M} Q^{c*}(s_i, \alpha_i) - Q^*(\boldsymbol{s}, \boldsymbol{a})| \leq \gamma \frac{M-1}{M(1-\gamma)} C_s \tag{24}$$

Proof : Let $Q_t^c(\cdot, \cdot)$ and $V^c(\cdot)$ be the results at the t^{th} iteration of the CW-MDP algorithm, we have:

$$\begin{aligned} \max_{\mathbf{s}} \max_{\mathbf{a}} |\sum_{i=1}^{M} Q_t^c(s_i, \alpha_i) - Q^*(\mathbf{s}, \mathbf{a})| = \max_{\mathbf{s}} \max_{\mathbf{a}} |\sum_i R_I(s_i, \alpha_i) + \gamma \sum_i \sum_{s_i'} T(s_i'|s_i, \alpha_i) V_t^c(s_i') \\ - R(\mathbf{s}, \mathbf{a}) - \gamma \sum_{\mathbf{s}'} T(\mathbf{s}'|\mathbf{s}, \mathbf{a}) V^*(\mathbf{s}')|, \end{aligned} \tag{25}$$

where $V^*(\mathbf{s})$ is the optimal global value function for state $\boldsymbol{s}$. Since $< \alpha_1, \alpha_2, ..., \alpha_M >$ is a valid combination of extended single-agent actions that correspond to the system-level action $\mathbf{a}$. Since the transition dynamics of the components are independent of each other, we have $T(s_i'|s_i, \alpha_i) = T(s_i'|\mathbf{s}, \mathbf{a})$ and

$$\begin{aligned} \sum_i \sum_{s_i'} T(s_i'|s_i, \alpha_i) V_t^c(s_i') = \sum_i \sum_{s_i'} T(s_i'|\mathbf{s}, \mathbf{a}) V_t^c(s_i') = \sum_i E(V_t^c(s_i')|\mathbf{s}, \mathbf{a}) \\ = \sum_{\mathbf{s}'} T(\mathbf{s}'|\mathbf{s}, \mathbf{a}) \sum_i V_t^c(s_i') \end{aligned} \tag{26}$$

Let $\boldsymbol{V_t^c}$ and $\boldsymbol{V^*} \in \mathbb{R}^{|\mathcal{S}|}$ be the vectors of $V_t^c(\mathbf{s})$ and $V^*(\mathbf{s})$ for all state $\boldsymbol{s}$, where $V_t^c(\mathbf{s}) = \sum_i V_t^c(s_i)$. Based on (25)–(26), we have

$$\begin{aligned}\max_{\mathbf{s}} \max_{\mathbf{a}} |\sum_{i=1}^{M} Q_t^c(s_i, \alpha_i) - Q^*(\mathbf{s}, \mathbf{a})| &= \max_{\mathbf{s}} \max_{\mathbf{a}} |\gamma \sum_{\mathbf{s}'} T(\mathbf{s}'|\mathbf{s}, \mathbf{a})[\sum_i V_t^c(s_i') - V^*(\mathbf{s}')]| \\ &\leq \gamma ||\boldsymbol{V_t^c} - \boldsymbol{V^*}||_\infty\end{aligned} \tag{27}$$

Let's define another operator $\mathcal{T}_G$ as:

$$V^*(\mathbf{s}) = \max_{\mathbf{a}}(Q^*(\mathbf{s}, \mathbf{a})) = \max_{\mathbf{a}} \Big(R(\mathbf{s}, \mathbf{a}) + \gamma \sum_{\mathbf{s}'} T(\mathbf{s}'|\mathbf{s}, \mathbf{a})[V^*(\mathbf{s}')] \Big) = (\mathcal{T}_G \boldsymbol{V^*})(\mathbf{s}) \tag{28}$$

Then an upper bound of $||\boldsymbol{V_t^c} - \boldsymbol{V^*}||_\infty$ can be obtained as:

$$||\boldsymbol{V_t^c} - \boldsymbol{V^*}||_\infty \leq ||\mathcal{T}_m \boldsymbol{V_{t-1}^c} - \mathcal{T}_G \boldsymbol{V_{t-1}^c}||_\infty + ||\mathcal{T}_G \boldsymbol{V_{t-1}^c} - \mathcal{T}_G \boldsymbol{V^*}||_\infty \tag{29}$$

We can rewrite $||\mathcal{T}_m \boldsymbol{V_t^c} - \mathcal{T}_G \boldsymbol{V_t^c}||_\infty$ using equation (9) and (26) as:

$$\begin{aligned}||\mathcal{T}_m \boldsymbol{V_t^c} - \mathcal{T}_G \boldsymbol{V_t^c}||_\infty &= \max_{\mathbf{s}} |\sum_i \max_{\alpha_i} Q_t^c(s_i, \alpha_i) - \max_{\mathbf{a}} \Big(R(\mathbf{s}, \mathbf{a}) + \sum_{\mathbf{s}'} T(\mathbf{s}'|\mathbf{s}, \mathbf{a})[\sum_i V_t^c(s_i')] \Big)| \\ &= \max_{\mathbf{s}} |\sum_i \max_{\alpha_i} Q_t^c(s_i, \alpha_i) - \max_{\mathbf{a}} \Big(\sum_i R_I(s_i, \alpha_i) + \gamma \sum_i \sum_{s_i'} T(s_i'|s_i, \alpha_i) V_t^c(s_i') \Big)| \\ &= \max_{\mathbf{s}} |\sum_i \max_{\alpha_i} Q_t^c(s_i, \alpha_i) - \max_{\mathbf{a}} \sum_i Q_t^c(s_i, \alpha_i)|\end{aligned} \tag{30}$$

Note that $\sum_i \max_{\alpha_i} Q_t^c(s_i, \alpha_i) = \max_{\boldsymbol{\alpha}: \alpha_i \in \mathcal{A}^e} \sum_i Q_t^c(s_i, \alpha_i)$, where $\boldsymbol{\alpha}$ is a vector of all possible valid and invalid combinations of extended single-component actions. Furthermore, we can rewrite $||\mathcal{T}_m \boldsymbol{V_t^c} - \mathcal{T}_G \sum_i \boldsymbol{V_t^c}||_\infty$ as:

$$||\mathcal{T}_m \boldsymbol{V_t^c} - \mathcal{T}_G \boldsymbol{V_t^c}||_\infty = \max_{\mathbf{s}} |\max_{\boldsymbol{\alpha}} \sum_i Q_t^c(s_i, \alpha_i) - \max_{\mathbf{a}} \sum_i Q_t^c(s_i, \alpha_i)| \tag{31}$$

We can obtain an upper bound for $\max_{\mathbf{s}} |\max_{\boldsymbol{\alpha}} \sum_i Q_t^c(s_i, \alpha_i) - \max_{\mathbf{a}} \sum_i Q_t^c(s_i, \alpha_i)|$ by analyzing the terms in $\max_{\boldsymbol{\alpha}} \sum_i Q_t^c(s_i, \alpha_i)$ and $\max_{\mathbf{a}} \sum_i Q_t^c(s_i, \alpha_i)$. First note that all α_i in $\boldsymbol{\alpha}$ can only be either $a_{0,0}$ or $a_{1,1}$. If all α_i in $\boldsymbol{\alpha}$ are equal to $a_{0,0}$, then $\boldsymbol{\alpha}$ is a valid combination of extended single-component actions corresponding to a system-level action $\boldsymbol{a}$. Therefore $\max_{\boldsymbol{\alpha}} \sum_i Q_t^c(s_i, \alpha_i) - \max_{\mathbf{a}} \sum_i Q_t^c(s_i, \alpha_i) = 0$. If at least one α_i in $\boldsymbol{\alpha}$ is equal to $a_{1,1}$, it can be seen that if we change all $\alpha_i = a_{0,0}$ in $\boldsymbol{\alpha}$ to $\alpha_i = a_{0,1}$, then $\boldsymbol{\alpha}$ becomes a valid combination of extended single-component actions corresponding to a system-level action $\boldsymbol{a}$.

Since changing each $\alpha_i = a_{0,0}$ in $\boldsymbol{\alpha}$ to $\alpha_i = a_{0,1}$ reduces the reward by $\frac{C_s}{M}$ and the number of changes is at most $M-1$, we have

$$||\mathcal{T}_m \boldsymbol{V_t^c} - \mathcal{T}_G \boldsymbol{V_t^c}||_\infty = \max_{\mathbf{s}} |\max_{\boldsymbol{\alpha}} \sum_i Q_t^c(s_i, \alpha_i) - \max_{\mathbf{a}} \sum_i Q_t^c(s_i, \alpha_i)| \leq \frac{M-1}{M} C_s \tag{32}$$

Further, we can bound $||\mathcal{T}_G \boldsymbol{V_t^c} - \mathcal{T}_G \boldsymbol{V^*}||_\infty$, using the fact that operator $\mathcal{T}_G$ is a contraction mapping(Puterman, 2014) as:

$$||\mathcal{T}_G \boldsymbol{V_t^c} - \mathcal{T}_G \boldsymbol{V^*}||_\infty \leq \gamma ||\boldsymbol{V_t^c} - \boldsymbol{V^*}||_\infty \tag{33}$$

From (32) and (33) we have

$$\begin{aligned} ||\boldsymbol{V_t^c} - \boldsymbol{V^*}||_\infty \leq & ||\mathcal{T}_m \boldsymbol{V_{t-1}^c} - \mathcal{T}_G \boldsymbol{V_{t-1}^c}||_\infty + ||\mathcal{T}_G \boldsymbol{V_{t-1}^c} - \mathcal{T}_G \boldsymbol{V^*}||_\infty \\ \leq & \frac{M-1}{M} C_s + \gamma ||\boldsymbol{V_{t-1}^c} - \boldsymbol{V^*}||_\infty \\ \leq & \frac{M-1}{M} C_s [\sum_{k=1}^{t} \gamma^{t-k}] + \gamma^t ||V_0^c - V^*||_\infty \end{aligned} \tag{34}$$

Therefore, we can write $\max_{\mathbf{s}} \max_{\mathbf{a}} |\sum_{i=1}^{M} Q_t^c(s_i, \alpha_i) - Q^*(\mathbf{s}, \mathbf{a})|$ as:

$$\max_{\mathbf{s}} \max_{\mathbf{a}} |\sum_{i=1}^{M} Q_t^c(s_i, \alpha_i) - Q^*(\mathbf{s}, \mathbf{a})| \leq \gamma \left(\frac{M-1}{M} C_s [\sum_{k=1}^{t} \gamma^{t-k}] + \gamma^t ||\boldsymbol{V_0^c} - \boldsymbol{V^*}||_\infty \right) \tag{35}$$

At $t \longrightarrow \infty$ we have

$$\begin{aligned} \max_{\mathbf{s}} \max_{\mathbf{a}} |\sum_{i=1}^{M} Q^{c*}(s_i, \alpha_i) - Q^*(\mathbf{s}, \mathbf{a})| = & \lim_{t \longrightarrow \infty} \max_{\mathbf{s}} \max_{\mathbf{a}} |\sum_{i=1}^{M} Q_t^c(s_i, \alpha_i) - Q^*(\mathbf{s}, \mathbf{a})| \\ \leq & \gamma \lim_{t \longrightarrow \infty} \left(\frac{M-1}{M} C_s [\sum_{k=1}^{t} \gamma^{t-k}] + \gamma^t ||\sum_i V_0^c - V^*||_\infty \right) \\ = & \gamma \frac{M-1}{M(1-\gamma)} C_s \end{aligned} \tag{36}$$

□

Let $\boldsymbol{V_t^{c'}} \in \mathbb{R}^{|\mathcal{S}_c|}$ be the vector notation for the $V^{c'}$-function at a given t^{th} iteration step of the ACW-MDP algorithm. And, $Q_t^{c'}(s_i, .)$ corresponds to the $Q^{c'}$-function estimated using $V_t^{c'}(s_i)$ at a given t^{th} iteration step. Further, let $bolt_\lambda(Q_t^{c'}(s_i, .))$ be a *Boltzman operator* defined as:

$$bolt_\lambda(Q_t^{c'}(s_i, .)) = \sum_{\alpha_i \in \mathcal{A}^e} \left(\frac{e^{\lambda Q_t^{c'}(s_i, \alpha_i)}}{\sum_{\alpha_i \in \mathcal{A}^e} e^{\lambda Q_t^{c'}(s_i, \alpha_i)}} Q_t^{c'}(s_i, \alpha_i) \right) \tag{37}$$

The $bolt_\lambda(.)$ operator for a given $\lambda > 0$ is not a non-expansive operator as it has the following property(Cesa-Bianchi et al., 2017):

$$|bolt_\lambda(Q_1^{c'}(s_i,.)) - bolt_\lambda(Q_2^{c'}(s_i,.))| \leq ||Q_1^{c'}(s_i,.) - Q_2^{c'}(s_i,.)||_\infty + \frac{2\log(3)}{\lambda} \tag{38}$$

Hence, for $bolt_\lambda(Q_t^{c'}(s_i,.))$, convergence to a fixed point cannot be guaranteed. Instead, we can show that $\boldsymbol{V_t^{c'}}$ eventually converges to a bounded region around $\boldsymbol{V^{c*}}$ as given in the following proposition.

Proposition 2. *As $t \longrightarrow \infty$, the value of $\boldsymbol{V_t^{c'}}$ converges to a bounded region around the optimal $\boldsymbol{V^{c*}}$ for a given λ:*

$$\lim_{t \longrightarrow \infty} ||\boldsymbol{V_t^{c'}} - \boldsymbol{V^{c*}}||_\infty \leq \frac{\log(3)}{\lambda(1-\gamma)}. \tag{39}$$

The proof of the above proposition is given in the supplementary material. Since our decision depends on the estimates of the $Q^{c'}$ function, we also show that a finite bound exists between $Q_t^{c'}(s_i,\alpha_i)$ and $Q^{c*}(s_i,\alpha_i)$ as $t \longrightarrow \infty$ based on the following proposition.

Proposition 3. *As $t \longrightarrow \infty$, the value of $Q_t^{c'}(s_i,\alpha_i)$ converges to a bounded region around the optimal $Q^{c*}(s_i,\alpha_i)$ for a given λ:*

$$\lim_{t \longrightarrow \infty} \max_{s_i} \max_{\alpha_i} |Q_t^{c'}(s_i,\alpha_i) - Q^{c*}(s_i,\alpha_i)| \leq \gamma \frac{\log(3)}{\lambda(1-\gamma)}. \tag{40}$$

The proof of the above proposition is given in the supplementary material. Similarly to Proposition 1, for ACW-MDP we can find a bound between the $\boldsymbol{Q^{c'}}$ and the optimal system level Q function $\boldsymbol{Q^*}$, as given in the following proposition.

Proposition 4. *For a given valid combination of extended single-agent actions $< \alpha_1, \alpha_2, ..., \alpha_M >$ corresponding to a system-level action $\boldsymbol{a}$, the bound between $\sum_{i=1}^{M} Q_t^{c'}(s_i,\alpha_i)$ for an optimal parameter λ^* and $Q^*(\boldsymbol{s},\boldsymbol{a})$ as $t \longrightarrow \infty$ is given by:*

$$\lim_{t \longrightarrow \infty} \max_{\boldsymbol{s}} \max_{\boldsymbol{a}} |\sum_{i=1}^{M} Q_t^{c'}(s_i,\alpha_i) - Q^*(\boldsymbol{s},\boldsymbol{a})| \leq \gamma \frac{M\log(3)}{\lambda^*(1-\gamma)} + \gamma \frac{M-1}{M(1-\gamma)} C_s. \tag{41}$$

The proof of the above proposition is given in the supplementary material.

The bounds in Proposition 2 and 3 can be visualized in Figure 2. The bounds also provide us with insights into the convergence of ACW-MDP. This shows that though the exact convergence of ACW-MDP cannot be guaranteed, convergence to a closed region

in space is always guaranteed, which implies that starting from any arbitrary $\boldsymbol{V_0^{c'}}$, we eventually go close to $\boldsymbol{V^{c*}}$ in the space.

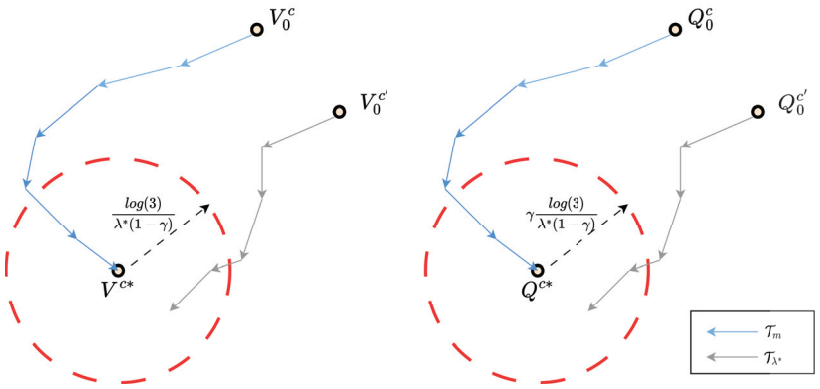

Figure 2: Visualization of convergence of single component of CW-MDP to a unique solution $\boldsymbol{V^{c*}}$ and $\boldsymbol{Q^{c*}}$, whereas single component of ACW-MDP converges to a bounded region around $\boldsymbol{V^{c*}}$ and $\boldsymbol{Q^{c*}}$

From bounds in Proposition 2 and 3, we can also observe that the solutions of ACW-MDP and CW-MDP will be close to each other for system settings that have large parameters λ^*. For systems with lower $\frac{C_s}{M}$ values, the difference between $Q^{c*}(s_i, a_{0,0})$ and $Q^{c*}(s_i, a_{0,1})$ would be small, allowing the max operator to be an equivalent choice. Also, in systems with very high $\frac{C_s}{M}$ value, components will get replaced simultaneously even if they are not close to failure, implying that the components will predominantly choose between action $a_{0,0}$ or $a_{1,1}$. System settings with exceptionally high and low values of $\frac{C_s}{M}$ tend to choose a higher value of λ^*. ACW-MDP solutions for such systems would be equivalent to CW-MDP solutions.

5 Numerical Studies

In this section, we conduct numerical studies to evaluate the proposed CW-MDP and ACW-MDP-based policies for CBM of large multi-component systems. For a large multi-component system, it is computationally infeasible to obtain an optimal system-level solution using methods like value iteration or MCTS. Therefore, we compare our proposed methods with existing heuristic policies: (n, N)-policy (Andersen, 2022) and (n, m, N)-policy(Anonymous, 2023).

The (n, N)-policy, $n \leq N$, for CBM of a system with *identical* components, can be

stated as follows: If one or more components degrades to a state greater than or equal to N, all components with state n or higher are replaced. The parameters n and N are chosen by performing a heuristic search on all $n, N \in \{1, 2, 3, ...L\}$. Similar to the (n, N)-policy, we can define the (n, m, N)-policy, $n \leq m \leq N$, as: If one or more components degrade to a state greater than or equal to N, or two or more components degrade to a state greater than or equal to m, then all components that are beyond the state n are replaced. The parameters n, m and N are chosen by performing a heuristic search over all $n, N \in \{1, 2, 3, ...L\}$.

In addition, we also compare the proposed methods with an independent component-wise policy. We solve the independent MDP for each component in the system, considering that the individual component receives an extra cost C_s/M whenever we choose action $a_i = 1$ (i.e., the component is replaced). The independent component-wise policy is equivalent to solving CW-MDP with actions $a_{0,0}$ and $a_{1,1}$ only. Comparison with an independent component-wise solution allows us to evaluate the effectiveness of using extended action space and system-level action selection for solving the CBM problem. We compare all our policies for 100 simulation steps by averaging the total discounted rewards for 10,000 trials with a discount factor $\gamma = 0.95$. In the numerical study, we use two different reward structures: $(C_m = 200; C_s = 1000; C_b = 1000)$ and $(C_m = 200; C_s = 1200; C_b = 1000)$. Initially, we analyze the performance of a system consisting of homogeneous components, i.e., all components in the system follow the same transition dynamics, i.e., $T_i = T \forall i$. Subsequently, we analyze the performance of a system consisting of heterogeneous components, i.e., all components in the system follow different transition dynamics. We can extend the (n, N) and (n, m, N)-policy to systems with heterogeneous components with the same values C_m, C_b, and C_s. We consider the same state space $\mathcal{S}_u$ for all heterogeneous components, allowing us to choose the same parameters n, m, and N for all components in the system. In both cases, we also perform a sensitivity analysis over setup cost C_s.

5.1 Homogeneous Component Systems

For a homogeneous component system, we consider a system with 20, 30, 40, 50, and 60 components, each component having the same state space $\mathcal{S}_u$ and transition dynamics. The transfer matrix is generated randomly using the assumption $T(s_i'|s_i, a_i) = 0$ if $s_i' < s_i$and $a_i = 0$; and $T(s_i'|s_i, a_i) \geq 0$ if $s_i' \geq s_i$, and $T(s_i^1|s_i, a_i) \geq T(s_i^2|s_i, a_i)$ if $s_i^2 \geq s_i^1 \geq s_i$. In

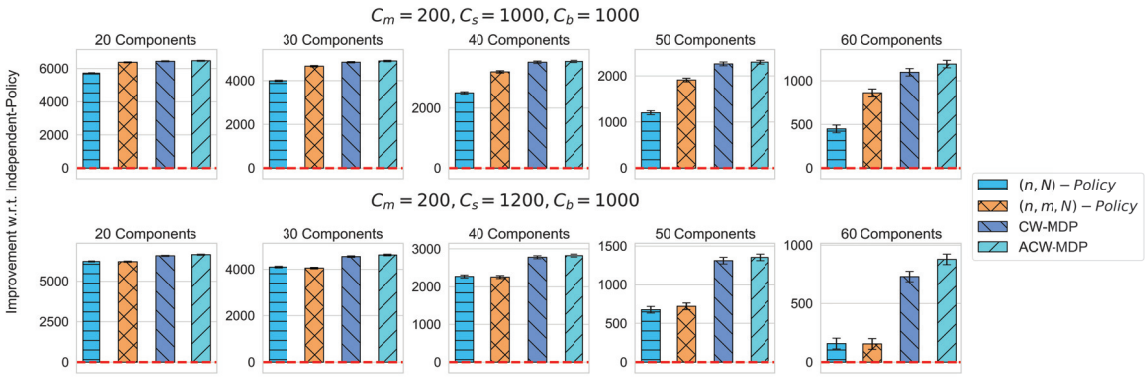

Figure 3: Bar plots showing improvement in average discount rewards with standard deviation(using whiskers) w.r.t independent component-wise solution for 20, 30, 40, 50, and 60 homogeneous components for (n, N)-policy, (n, m, N)-policy, CW-MDP, and ACW-MDP using two different reward structures

Figure 3, we show an improvement in the average discount rewards w.r.t independent component-wise solution for 20, 30, 40, 50, and 60 homogeneous components.

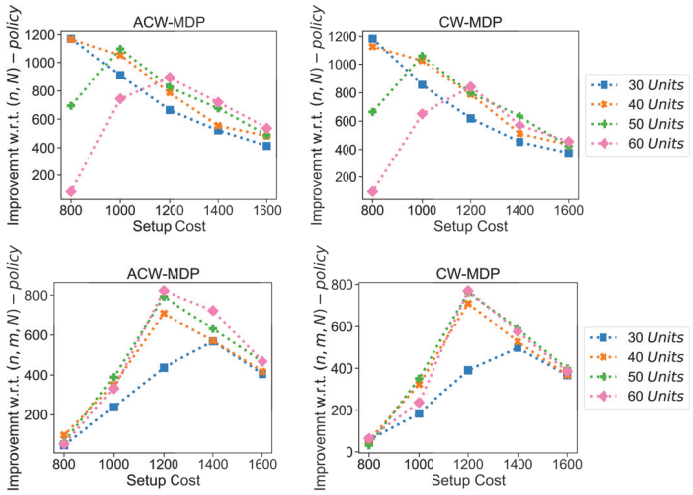

Figure 4: Improvement in discounted rewards for homogeneous components from CW-MDP and ACW-MDP w.r.t. (n, N) and (n, m, N)-policy with changing setup cost

For a system with 20 homogeneous components, we observe that under the reward structure $C_m = 200; C_s = 1000; C_b = 1000$, the performance of ACW-MDP and CW-MDP is close to that of (n, m, N)-policy. When we increase the setup cost to $C_s = 1200$ with the same corrective replacement cost $C_b = 1000$, we observe that CW-MDP and ACW-MDP perform better than the (n, N) and (n, m, N) policies. For systems with 30,

40, 50, and 60 components, we observe that our proposed CW-MDP and ACW-MDP approach performs better than the (n, N) and (n, m, N) policy. Under all cost structures, ACW-MDP performs equivalently or better than CW-MDP. Furthermore, the independent component-wise solution performs worse in comparison to all other policies. This is evident since independent solutions do not consider the impact of setup cost on component-wise maintenance decisions.

Since the setup cost C_s has a significant impact on the maintenance decisions of the components due to the induced positive economic dependence, we performed a sensitivity analysis to understand the impact of the setup cost C_s on the performance of CW-MDP and ACW-MDP. In Figure 4, we plot the improvement in total discounted rewards from CW-MDP and ACW-MDP w.r.t. (n, N) and (n, m, N)-policy for 30, 40, 50 and 60 component systems. For $C_s = 800$ the performance of both ACW-MDP and CW-MDP is close to that of (n, m, N). However, under the same setup cost $C_s = 800$, the improvement in discounted rewards w.r.t. (n, N)-policy decreases with the number of components in the system. This behavior is because, under the specific cost structure, CW-MDP and ACW-MDP perform comparably to the (n, m, N) policy. However, As the number of components increases, the impact of economic dependence between components decreases, leading to inferior performance of the (n, N) policy when compared to CW-MDP and ACW-MDP. Furthermore, on increasing setup cost, we observe that the performance of both ACW-MDP and CW-MDP maximizes in a region of the setup cost. At a higher setup cost, components tend to be replaced together even if they are close to failure; this makes (n, m, N) and (n, N)-policy to perform close to ACW-MDP and CW-MDP.

5.2 Heterogeneous Component Systems

For a heterogeneous component system, we consider a system with 20, 30, 40, 50, and 60 components, each component having the same state space $\mathcal{S}_u$ but different transition dynamics T_i for each component. This choice is motivated by the fact that the existing heuristic policies, such as (n, N)-Policy and (n, m, N)-Policy, are typically designed for systems where components have the same degradation level L. We generate a random transition matrix for the problem using the assumption $T(s_i'|s_i, a_i) = 0$ if $s_i' < s_i$and $a_i = 0$; and $T(s_i'|s_i, a_i) \geq 0$ if $s_i' \geq s_i$, and $T(s_i^1|s_i, a_i) \geq T(s_i^2|s_i, a_i)$ if $s_i^2 \geq s_i^1 \geq s_i$.

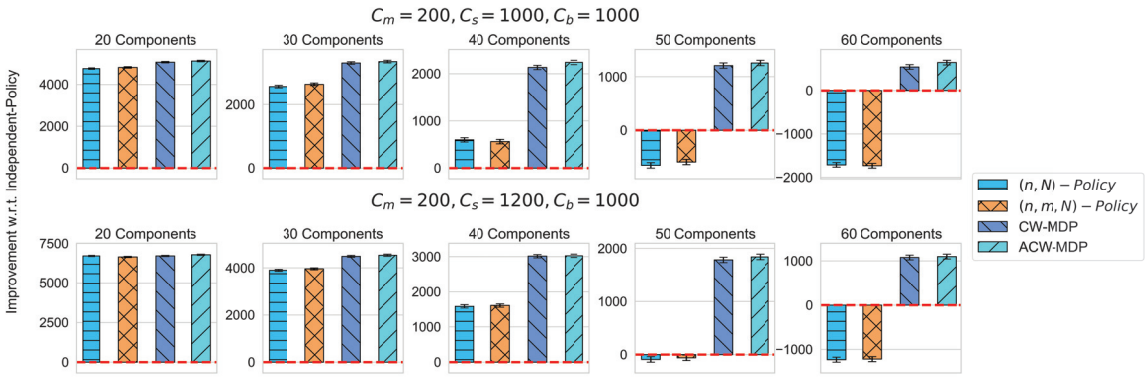

Figure 5: Bar plots showing improvement in average discount rewards with standard deviation(using whiskers) w.r.t independent component-wise solution for 20, 30, 40, 50, and 60 heterogeneous components for (n, N)-policy, (n, m, N)-policy, CW-MDP, and ACW-MDP using two different reward structures

In Figure 5, we show an improvement in the average discount rewards w.r.t independent component-wise solution for 20, 30, 40, 50, and 60 heterogeneous components. Under both reward structures for systems with 20, 30, 40, 50, and 60 heterogeneous components, our proposed ACW-MDP and CW-MDP perform better than (n, N), (n, m, N). Specifically, for systems with 50 and 60 heterogeneous components, the (n, N) and (n, m, N)-policy performs worse than the independent component-wise solution. Also, ACW-MDP always performs either equivalently or better than CW-MDP. The improvement in the policies generated using ACW-MDP and CW-MDP w.r.t. (n, N) and (n, m, N)-policy increases with the number of components in the system. A higher number of components in the system implies a higher heterogeneity in the system, leading to poor performance of heuristic policies.

Similar to the homogeneous component systems, we perform a sensitivity analysis for heterogeneous component systems to understand the impact of setup cost C_s on the performance of CW-MDP and ACW-MDP. In Figure 6, we plot the improvement in the total discount rewards of CW-MDP and ACW-MDP w.r.t. (n, N) and (n, m, N)-policy for 30, 40, 50 and 60 heterogeneous component systems. In contrast to the homogeneous case, in heterogeneous systems, improvements do not peak in a region of setup cost. Instead, we observe an almost significant improvement in total discounted rewards of ACW-MDP and CW-MDP w.r.t. (n, N) and (n, m, N)-policy on changing the setup cost. This happens

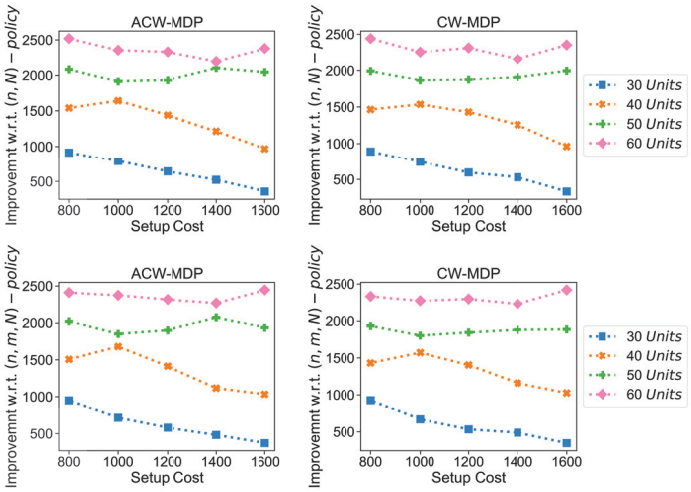

Figure 6: Improvement in Discounted Rewards for heterogeneous components from CW-MDP and ACW-MDP w.r.t. (n, N) and (n, m, N)-policy with changing setup cost

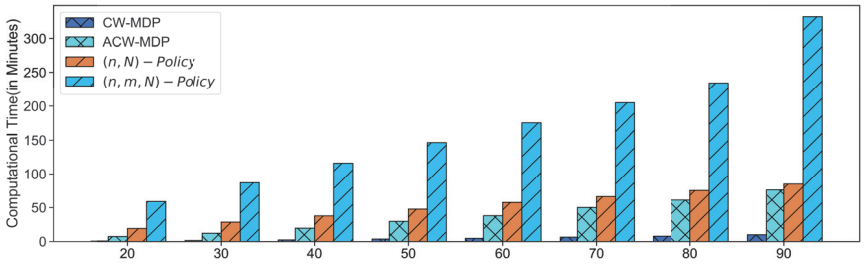

Figure 7: Computational time(in minutes) for CW-MDP, ACW-MDP, (n, N) and (n, m, N)-policy for Heterogeneous component system for 10,000 trials

due to the different transition dynamics of each component, allowing each component to be impacted differently by the setup cost in the system. Some components tend to fail faster compared to other components that require replacement more often, irrespective of the setup cost. Also, we observe that magnitude of improvement increases with the number of components in the system.

Figure 7 shows a comparison of the computational time for the CW-MDP, ACW-MDP, (n, N) and (n, m, N)-policy for heterogeneous component system for 10,000 trials to evaluate a policy for 100 simulation steps with $L = 10$. We observe that CW-MDP is highly efficient compared to all other methods since it requires solving an MDP of state space size $|\mathcal{S}_u| = L$ and action space $|\mathcal{A}_e| = 3$. Further, the computational time of ACW-MDP is higher than CW-MDP because we iteratively evaluate our policy at the system level to find parameter λ.

Furthermore, the (n, m, N)-policy has the highest computational time because it requires a huge heuristic search for all combinations of n, m, and N. For each combination, we need to evaluate policy at the system level making (n, m, N)-policy highly computationally expensive to estimate.

Additionally, We can analyze the computational complexity of the method by making the assumption that both methods require n iterations to converge. For n iterations, CW-MDP would require $3nL$ computational steps. Similarly, for ACW-MDP, assuming a search space $\mathcal{L}$ of size k for λ^*, and considering n iterations, it would require $3nkL$ computational steps. Additionally, ACW-MDP involves evaluating individual λ values by simulating the system to find the optimal λ^*. We can see that the computational overhead of ACW-MDP compared with CW-MDP is mainly determined by the tuning efforts for the parameter λ^*. It is important to note that, in theory, ACW-MDP can provide a solution equivalent to or better than CW-MDP. Therefore, it is beneficial to explore a certain set of values $\mathcal{L}$ to check if there exists a λ^* that can improve the CBM policies.

5.3 Case Study: Maintenance of Wind Turbine Bearings

In this section, we show an application of our proposed ACW-MDP and CW-MDP to develop a maintenance policy for a large multi-component system consisting of a gearbox bearing for multiple wind turbines. The gearbox is a crucial component of a wind turbine used to increase the rotational speed of a low-speed rotor to a high-speed electrical generator. Gearbox bearings are usually subjected to wear due to external and internal factors. Failure of a gearbox can lead to failure of wind turbines and would lead to massive operational and equipment loss. Timely maintenance of the gearbox can prevent these failures and allow efficient functioning of wind turbines. We consider a gearbox-bearing system consisting of 20 to 150 homogeneous components. The degradation of the gearbox is modeled using a discrete state degradation process with $L = 4$ for which we consider a transition matrix T as given in (Li et al., 2019) defined as:

$$\mathrm{T} = \begin{bmatrix} 0.8571 & 0.1429 & 0 & 0 \\ 0 & 0.8571 & 0.1429 & 0 \\ 0 & 0 & 0.8 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{42}$$

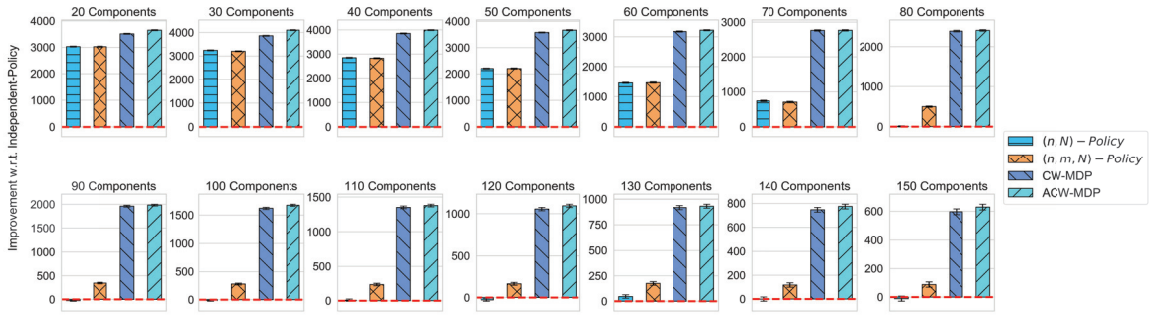

Figure 8: Bar plots showing improvement in average discount rewards with standard deviation(using whiskers) w.r.t independent component-wise solution for systems with a large number of bearings for (n, N)-policy, (n, m, N)-policy, CW-MDP, ACW-MDP

In the above matrix, the $(i, j)^{th}$ element, $T_{i,j}$, corresponds to the probability of transitioning from a state i to a state j. We consider a reward structure with $C_m = 200, C_s = 800$, and $C_b = 1000$. We evaluate all policies by finding the average total discounted rewards for 100 simulation steps with a discount factor $\gamma = 0.95$, averaging over 10,000 trials. In Figure 8, we show improvement in average discount rewards w.r.t independent component-wise solution for systems with 20 to 150 bearings using different policies.

We observe that for system size ranging from 20 to 150, our proposed ACW-MDP and CWMDP perform better than (n, N), (n, m, N) and independent component-wise solution. Additionally, when the number of components is greater than 80, we observe that the (n, N)-policy performs equivalent to an independent component-wise solution. This happens because of a fixed setup cost; by increasing the number of components in the system, the impact of positive economic dependence decreases. Also, the ACW-MDP always performs better or is equivalent to CW-MDP. We can also note that, as the number of components increases, the rewards from ACW-MDP start to come closer to CW-MDP. This happens as increasing the number of components decreases the value of $\frac{C_s}{M}$ allowing the system to choose a higher λ^* making ACW-MDP equivalent to CW-MDP.

6 Conclusion

In this work, we proposed two different methods, CW-MDP and ACW-MDP, for developing a CBM policy for large multi-component systems. Using extensive experimental studies, we show that our proposed methods perform better than the existing heuristic policies,

namely, (n, N) and (n, m, N)-policy. We observe that our proposed policies give a large improvement in total discounted rewards compared to the policy (n, N) and (n, m, N) for both homogeneous and heterogeneous component systems. We also studied the impact of setup cost on ACW-MDP and CW-MDP and compared improvements with respect to (n, N) and (n, m, N)-policy. For a homogeneous system, we observe that for lower setup-cost our methods give similar performance w.r.t. (n, m, N)-policy. However, with increasing setup cost, we observe that CW-MDP and ACW-MDP perform better than the (n, N) and (n, m, N)-policy. For a heterogeneous system, we observe a consistent improvement in performance using ACW-MDP and CW-MDP w.r.t., (n, N) and (n, m, N)-policy. In addition, the scale of improvements increases with the number of components in the system. Additionally, we observe that ACW-MDP and CW-MDP are computationally efficient in comparison to (n, N) and (n, m, N)-policy, allowing us to scale our method for large multi-component systems. Furthermore, we analyzed the convergence and relationship between the solutions of ACW-MDP and CW-MDP. Though the component-wise solution does not converge to the exact optimal system-level solution, it allows us to solve problems that are computationally infeasible.

The future research direction for this work is to extend the component-wise solutions for more complex system dependence structures such as series-parallel systems or k-out-of-n systems. Another major limitation of this work is that we consider a uniform distribution of setup cost among the components. In the future, we will work on developing methods that can consider the non-uniform distribution of setup costs or more complex reward structures with multiple setup costs in heterogeneous systems.

7 Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon request.

References

Alaswad, S. and Xiang, Y. (2017). A review on condition-based maintenance optimization models for stochastically deteriorating system. *Reliability engineering & system safety*, 157:54–63.

Andersen, J. F. (2022). Maintenance optimization for multi-component systems using markov decision processes.

Andersen, J. F., Andersen, A. R., Kulahci, M., and Nielsen, B. F. (2022). A numerical study of markov decision process algorithms for multi-component replacement problems. *European Journal of Operational Research*, 299(3):898–909.

Anonymous (2023). A rollout approach for condition-based maintenance of large multi-unit systems. *IEEE Transaction on Reliability, submitted*.

Barata, J., Soares, C. G., Marseguerra, M., and Zio, E. (2002). Simulation modelling of repairable multi-component deteriorating systems for 'on condition'maintenance optimisation. *Reliability Engineering & System Safety*, 76(3):255–264.

Barbera, F., Schneider, H., and Watson, E. (1999). A condition based maintenance model for a two-unit series system. *European Journal of Operational Research*, 116(2):281–290.

Bouvard, K., Artus, S., Bérenguer, C., and Cocquempot, V. (2011). Condition-based dynamic maintenance operations planning & grouping. application to commercial heavy vehicles. *Reliability Engineering & System Safety*, 96(6):601–610.

Cesa-Bianchi, N., Gentile, C., Lugosi, G., and Neu, G. (2017). Boltzmann exploration done right. *Advances in neural information processing systems*, 30.

Dehghani Ghobadi, Z., Haghighi, F., and Safari, A. (2022). An overview of reinforcement learning and deep reinforcement learning for condition-based maintenance. *International Journal of Reliability, Risk and Safety: Theory and Application*.

Keizer, M. C. O., Flapper, S. D. P., and Teunter, R. H. (2017a). Condition-based maintenance policies for systems with multiple dependent components: A review. *European Journal of Operational Research*, 261(2):405–420.

Keizer, M. C. O., Teunter, R. H., and Veldman, J. (2017b). Joint condition-based maintenance and inventory optimization for systems with multiple components. *European Journal of Operational Research*, 257(1):209–222.

Kochenderfer, M. J., Wheeler, T. A., and Wray, K. H. (2022). *Algorithms for decision making*. MIT press.

Koochaki, J., Bokhorst, J. A., Wortmann, H., and Klingenberg, W. (2013). The influence of condition-based maintenance on workforce planning and maintenance scheduling. *International Journal of Production Research*, 51(8):2339–2351.

Li, J., Zhang, X., Zhou, X., and Lu, L. (2019). Reliability assessment of wind turbine bearing based on the degradation-hidden-markov model. *Renewable energy*, 132:1076–1087.

Littman, M. L. and Szepesvári, C. (1996). A generalized reinforcement-learning model: Convergence and applications. In *ICML*, volume 96, pages 310–318.

Liu, X., Li, J., Al-Khalifa, K. N., Hamouda, A. S., Coit, D. W., and Elsayed, E. A. (2013). Condition-based maintenance for continuously monitored degrading systems with multiple failure modes. *IIE transactions*, 45(4):422–435.

Liu, Y., Chen, Y., and Jiang, T. (2020). Dynamic selective maintenance optimization for multi-state systems over a finite horizon: A deep reinforcement learning approach. *European Journal of Operational Research*, 283(1):166–181.

Marseguerra, M., Zio, E., and Podofillini, L. (2002). Condition-based maintenance optimization by means of genetic algorithms and monte carlo simulation. *Reliability Engineering & System Safety*, 77(2):151–165.

Peng, S. et al. (2021). Reinforcement learning with gaussian processes for condition-based maintenance. *Computers & Industrial Engineering*, 158:107321.

Pinciroli, L., Baraldi, P., and Zio, E. (2023). Maintenance optimization in industry 4.0. *Reliability Engineering & System Safety*, page 109204.

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Wang, R. and Chen, N. (2016). A survey of condition-based maintenance modeling of multi-component systems. In *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 1664–1668. IEEE.

Wong, A., Bäck, T., Kononova, A. V., and Plaat, A. (2023). Deep multiagent reinforcement learning: Challenges and directions. *Artificial Intelligence Review*, 56(6):5023–5056.

Xia, T., Xi, L., Zhou, X., and Lee, J. (2013). Condition-based maintenance for intelligent monitored series system with independent machine failure modes. *International Journal of Production Research*, 51(15):4585–4596.

Xu, J., Zhao, X., and Liu, B. (2022). A risk-aware maintenance model based on a constrained markov decision process. *IISE Transactions*, 54(11):1072–1083.

Yousefi, N., Tsianikas, S., and Coit, D. W. (2022). Dynamic maintenance model for a repairable multi-component system using deep reinforcement learning. *Quality Engineering*, 34(1):16–35.

Zhang, N. and Si, W. (2020). Deep reinforcement learning for condition-based maintenance planning of multi-component systems under dependent competing risks. *Reliability Engineering & System Safety*, 203:107094.

Zhang, X. and Zeng, J. (2015). Deterioration state space partitioning method for opportunistic maintenance modelling of identical multi-unit systems. *International Journal of Production Research*, 53(7):2100–2118.

Zhang, Z., Wu, S., and Li, B. (2011). A condition-based and opportunistic maintenance model for a two-unit deteriorating system. In *2011 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering*, pages 590–595. IEEE.

Zhao, X., Gaudoin, O., Doyen, L., and Xie, M. (2019). Optimal inspection and replacement policy based on experimental degradation data with covariates. *IISE Transactions*, 51(3):322–336.