Multi-component Maintenance Optimization: A Stochastic Programming Approach

Zhicheng Zhu[1], Yisha Xiang[1], Bo Zeng[2]

[1]Department of Industrial Engineering, Lamar University, Beaumont, TX 77710
[2]Department of Industrial Engineering, The University of Pittsburgh, Pittsburgh, PA 15261

Abstract

Maintenance optimization has been extensively studied in the past decades. However, most of the existing maintenance models focus on single-component systems. Multi-component maintenance optimization, which joins the stochastic failure processes with the combinatorial maintenance grouping problems, remains as an open issue. To address this challenge, we study this problem in a finite planning horizon by i) developing a set of novel modeling techniques and building a two-stage stochastic integer model, and ii) based on its structural properties, designing and implementing an efficient heuristic algorithm under the progressive hedging framework. Comparing to three popular methods for stochastic integer programming, our algorithm demonstrates a drastically improved capacity in handling practically large-size problems. By using a rolling horizon scheme, our approach is further benchmarked with a conventional dynamic programming approach adopted in the literature. Numerical results show that the stochastic maintenance model and the designed heuristic can lead to significant cost savings.

Key words: maintenance optimization, multi-component system, stochastic programming, progressive hedging algorithm, heuristic

1. Introduction

Effective maintenance plays an important role in maintaining high levels of productivity and safety in many capital-intensive industries, especially those that operate complex, hazardous systems, such as offshore oil and gas drilling systems, nuclear power plants, petrochemical plants, and space transport systems (Alkhamis and Yellen 1995, Cowing et al. 2004, Laggoune et al. 2009). A number of catastrophic failures, e.g., the space shuttle Challenger accident (Alkhamis and Yellen 1995) and the loss of Piper Alpha oil platform (Paté‐Cornell 1993), have occurred in part because of inadequate maintenance. Moreover, the downtime cost caused by either planned or unplanned maintenance shutdown in these industries is often significant. The production losses can range from $5,000 to $100,000 per hour during the shutdown in chemical plants and millions of dollars per day in offshore drilling/refineries (Tan and Kramer 1997, Amaran et al. 2016). As the demand for high reliability increases, it is more imperative to develop efficient maintenance schedules for complex systems.

Maintenance optimization has been extensively studied in the literature. However, most of the existing maintenance models focus on single-component systems, and are not applicable for complex systems consisting of multiple components, because of various interactions between the components. In general, there are three different types of interactions: economic, structural, and stochastic dependence (Thomas 1986). Economic dependence is the most common one among these three types of interactions. Systems with the economic dependence typically incur a common system cost, the so-called setup cost, due to mobilizing repair crew, safety provisions, disassembling machines, special transportation, and the downtime loss. These costs are shared by all maintenance activities performed simultaneously. Considerable cost savings can be obtained

by jointly maintaining several components instead of separately, especially when the setup cost is high.

Multi-component maintenance problem is challenging in both modeling and solution techniques, and has remained as an open issue in the literature. This is because multi-component maintenance planning joins the stochastic processes regarding the failures of the components with the combinatorial problems regarding the grouping of maintenance activities (Dekker et al. 1997, Scarf 1997, Dekker and Scarf 1998, Van Horenbeek and Pintelon 2013). The problem can quickly end in complex models and explicit analytical expressions for optimal maintenance costs and the corresponding decisions are sometimes impossible to derive. One often has to make special system assumptions (Castanier et al. 2005, Tian and Liao 2011, Huynh et al. 2015), restrict grouping policies (Okumoto and Elsayed 1983, Assaf and Shanthikumar 1987, Archibald and Dekker 1996, Dekker et al. 1996, Dekker et al. 1997, Wildeman and Dekker 1997, Wildeman et al. 1997, Pham and Wang 2000, Rao and Bhadury 2000, Cui and Li 2006, Besnard et al. 2009, Ponchet et al. 2010, Bouvard et al. 2011, Ding and Tian 2012, Koochaki et al. 2012, Van Horenbeek and Pintelon 2013, Vu et al. 2014, Nguyen et al. 2015, Zhu et al. 2015), and/or resort to simulation tools (Tan and Kramer 1997, Bérenguer et al. 2000, Barata et al. 2002, Laggoune et al. 2009, Nguyen et al. 2015) so that the decision problem can be formulated with less mathematical difficulty.

A common approach to coordinating maintenance activities of multiple components is to partition the components into a number of fixed groups and then always maintain the components in a group jointly (van Dijkhuizen and van Harten 1996). This approach is also referred to as direct-grouping. The problem formulated with this approach is an NP-complete set-partitioning problem. The optimal grouping decision can be found for only a small number of

components due to the computational complexity. There are some efforts that reduce the set-partitioning problem for multi-component maintenance to a dynamic-programming problem with a quadratic time complexity in (Dekker et al. 1996, Wildeman et al. 1997). However, some special assumptions are made to allow such a reduction.

There are a series of papers that conduct direct grouping in a rolling horizon and allow incorporation of dynamic information (Dekker et al. 1996, Wildeman et al. 1997, Van Horenbeek and Pintelon 2013, Vu et al. 2014). But the grouping phase in these papers is still essentially a set-partitioning problem which is resolved in the rolling horizon if the situation variable (e.g., usage of components and environmental conditions) changes. A major deficiency in this series of papers is that each component is preventively maintained only one time within the planning horizon, which is often determined by the maximum replacement interval of individual components. This assumption is not relevant since a system may be composed of different components with different lifetime cycles, and maintenance intervals of components can be significantly different (Laggoune et al. 2009).

In contrast to the direct-grouping that yields a fixed group structure, an indirect grouping strategy groups preventive maintenance (PM) activities by making the PM interval a multiple of a basis interval, so the maintenance of different components can coincide (Goyal and Kusy 1985, Goyal and Gunasekaran 1992, Vos de Wael 1995). Another indirect grouping strategy performs major PM on all components jointly at the end of a common interval and allows minor or major PM within this interval. Indirect grouping model of this kind is sometimes formulated as a mixed integer programing (MIP) problem (Sule and Harmon 1979, Epstein and Wilamowsky 1985, Hariga 1994). Because of the simplified policy structure, the MIP model can be separated by components, which greatly reduces the computational complexity.

Both direct- and indirect-grouping focus on grouping PM activities, and ignore maintenance opportunities generated by corrective maintenance (CM) at failures. To take advantage of the shutdown due to some component's failure and use it as opportunities for PM of other functioning components, various opportunistic maintenance (OM) models have been proposed (Pham and Wang 2000, Rao and Bhadury 2000, Cui and Li 2006, Besnard et al. 2009, Ding and Tian 2012, Koochaki et al. 2012, Patriksson et al. 2015). The failure arrival process under OM is also mathematically complex. Simulation method is used to evaluate the cost function of OM models and simulation-based optimization methods are used to find optimal policies (Ding and Tian 2012, Koochaki et al. 2012). Shafiee and Finkelstein (2015) provide an analytical expression of the cost function under the rather simplified OM policy that preventively replaces all non-failed components when there is a failure. More recently, Patriksson et al. (2015) apply a stochastic programming approach in OM. The integer L-shaped method used in their model becomes prohibitive when the number of components is large.

Most of the OM models do not consider grouping opportunities presented by PM activities. One exception is the condition-based OM model proposed by Castanier et al. (2005). They first define an inspection/replacement policy for each component. Upon inspection, if the deterioration level of a component is between the PM and the failure thresholds, PM is performed. PM of one component presents an opportunity for other components with deterioration level between the OM and PM thresholds. The problem is formulated as a semi-regenerative process. This approach similarly suffers the computational intractability, because the problem size grows exponentially as the number of components increases. As a result, their analysis is limited to a two-component system (Castanier et al. 2005, Huynh et al. 2015). For

more details regarding the multi-component maintenance problem, the readers are referred to review papers (Thomas 1986, Dekker et al. 1997, Nicolai and Dekker 2008).

Our review of the literature shows that few research has considered grouping at both preventive and corrective maintenance occasions under practical assumptions, which significantly affects the optimality of the solutions because of the simplified models and reduced solution space. There is also a lack of efficient algorithms that can provide satisfactory results for practically large-scale multi-component maintenance problems.

To meet these needs, we develop a more general multi-component maintenance optimization problem in a finite-time horizon in this paper. We do not pose any restriction on the types of maintenance activities that can be grouped or when the grouping can occur. In other words, joint execution of any combination of maintenance activities can occur anytime. A novel two-stage stochastic linear integer model is formulated and an efficient algorithm for practical-size large problems is designed. Computational studies are conducted to illustrate the proposed algorithm's capability of solving large-scale problems. To assess the potential benefits from the proposed stochastic programming approach, we further compare the results of our two-stage stochastic maintenance model with those of a direct-grouping model using a dynamic-programming approach (Wildeman et al. 1997) over the rolling horizon. The main contributions of this paper are as follows.

(1) From a modeling perspective, this work extends the multi-component maintenance literature by using a stochastic programming approach. We develop an innovative set of modeling techniques and build a two-stage linear integer model. Comparing to existing modeling tools, this stochastic programming approach allows us to model multi-component maintenance planning for a generic system without posing any special system

assumption or grouping policy restrictions. This new modeling capacity largely increases the solution space and can lead to substantial cost savings. The adoption of stochastic programming approach further facilitates the derivation of analytical expressions for the total cost function and maintenance decisions and enables the use of stochastic programing optimization tools. The modeling techniques developed and demonstrate in this work are strong and innovative, and opens new research and implementation opportunities.

(2) From a solution technique perspective, based on critical structural properties, we design and implement an efficient heuristic algorithm under the progressive hedging framework. Comparing to three popular methods for stochastic integer programming, our algorithm demonstrates a drastically improved capacity in computing practically large-size problems. By using a rolling horizon scheme, our approach is further benchmarked with a conventional dynamic programming approach adopted in the literature. Numerical results show that the stochastic maintenance model and the designed heuristic can lead to significant cost savings.

The remainder of this paper is organized as follows. In section 2, we develop the deterministic extensive form (DEF) of proposed model. Section 3 describes the progressive-hedging-based heuristic algorithm and three benchmark algorithms in detail. Computational studies are presented in Section 4. Section 5 conducts sensitivity analysis to assess the potential benefits of the proposed model solved by our heuristic algorithm in the rolling horizon. We conclude this study and discuss future research directions in section 6.

2. Model development

Notation

Parameters	
n	Number of components
N	Component set
T	Length of the planning horizon
T_S	Time set
q	Number of individuals
R	Individual set
Ω	Set of scenarios
I_{ir}	Individual r of component i
T_{ir}^{ω}	Lifetime of I_{ij} in scenario ω
T'	Extended planning horizon, $T' = \max_{i,r,\omega} T_{ir}^{\omega}$
$c_{i,\mathrm{PR}}$	Preventive replacement (PR) cost for type i component's individual
$c_{i,\mathrm{CR}}$	Corrective replacement (CR) cost for type i component's individual
$C_{i,\mathrm{PR}}$	Total PR cost incurred by individuals of component i in the planning horizon [0, T]
$C_{i,\mathrm{CR}}$	Total CR cost incurred by individuals of component i in the planning horizon [0, T]
C_s	Total setup cost in the planning horizon [0, T]
d	Setup cost
s	Current time
δ	Length of a decision period
ξ_i	Equal to 1 when the working individual of component i is failed at the current time, otherwise 0
$\boldsymbol{\xi}$	Vector of working individuals' states at the current time
$Q(\mathbf{x})$	Expected second-stage cost
$p(\omega)$	Probability that scenario ω occurs
First-Stage Decision Variables	
x_i	Equal to 1 when the individual of component i at the current time is replaced, 0 otherwise
z	Equal to 1 when there is at least one individual maintained at current time, 0 otherwise
Second-Stage Decision Variables	
$\tilde{x}_{it}^{r\omega}$	Equal to 1 when I_{ir} is replaced at or before time t in scenario ω, 0 otherwise
z_t^{ω}	Equal to 1 when there is at least one individual maintained at time t in scenario ω, 0 otherwise

2.1 System Description

Consider a system that consists of $N = \{1, \ldots, n\}$ components. Each component in the system is considered as a different type of component regardless of its physical type. A system with a total of n components therefore has n types of components. To distinguish the component type and the component itself, we use component only when referring to its type and refer to physical components as individuals. For example, individual I_{ir} is the individual used for the r^{th} replacement of the type i component.

The costs of preventive and corrective replacements are $c_{i,\text{PR}}$ and $c_{i,\text{CR}}$, respectively. The preventive replacement (PR) cost is lower than the corrective replacement (CR) cost, $c_{i,\text{PR}} < c_{i,\text{CR}}$. The system setup cost is d at any maintenance occasion regardless of the number of individuals replaced. If n individuals are replaced at the same time, the total savings from executing these n maintenance activities jointly is $d(n - 1)$.

2.2 Two-stage Stochastic Programming Model

In this section, we develop a two-stage stochastic maintenance optimization model with the objective of minimizing the total expected cost in the planning horizon. The here-and-now decision is to select a group of individuals for PR at the current decision time. The wait-and-see decision is to determine groups of individuals for PR at future decision times. Since each t is a decision stage, this is a multi-stage decision-making problem. In this study, we use a two-stage model to approximate the multi-stage model by combining all future decision times after the current time in the second stage. We model the lifetime of each component with an appropriate distribution. For each component, we randomly generate lifetimes of all its individuals. A combination of lifetimes of all individuals of all component types is referred to as a scenario. For the first individual of any component at the current time, its lifetime is the remaining lifetime.

Notice that the lifetimes of individuals are deterministic for a given scenario using this scenario-generation method.

Before describing the details of model development, we first use a two-component system to illustrate how savings can be obtained by grouping some maintenance activities in a one-scenario

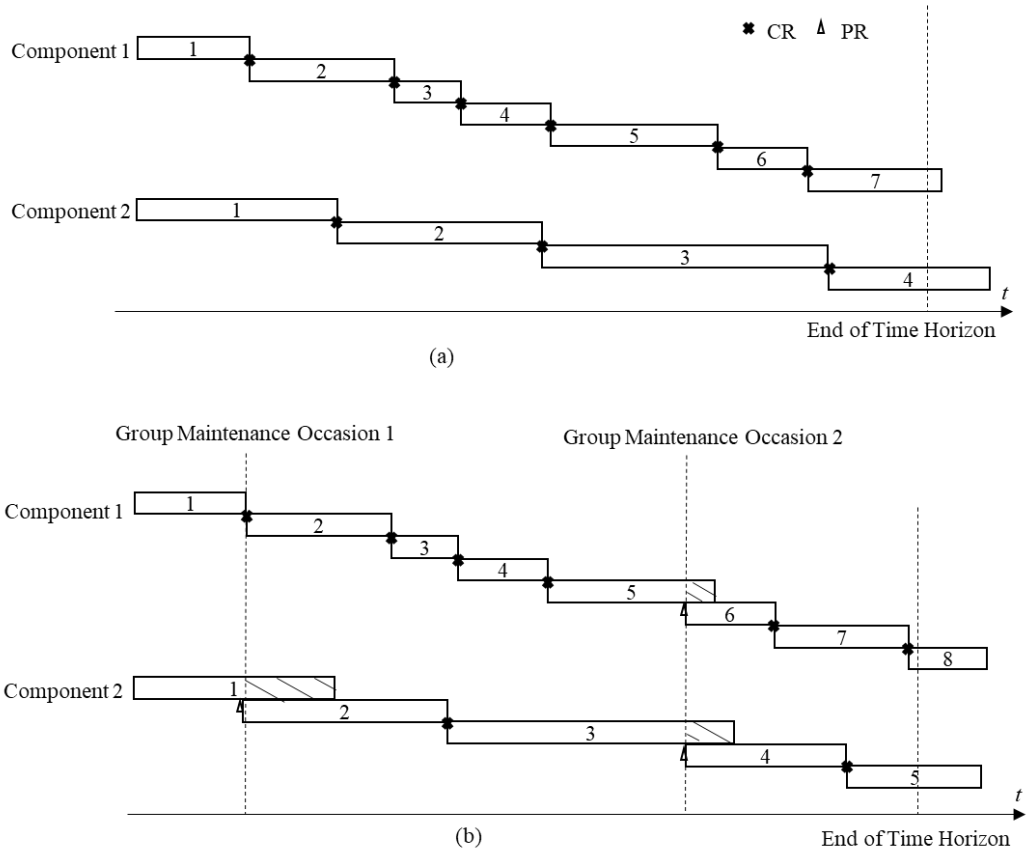

Figure 1: Illustration of possible group structures in one

problem setting. In Figure 1, each individual is represented by one bar whose length represents the lifetime of the corresponding individual. Figure 1(a) shows the group structure that replaces every individual upon failure, meaning no planned grouping activity. Figure 1(b) provides a group structure with some grouping activities. At group maintenance occasion 1 in Figure 1(b), individual $I_{2,1}$ is preventively replaced when $I_{1,1}$ is correctively replaced. The shaded area in $I_{2,1}$ is its not-used-residual lifetime. The saving in the setup cost at group maintenance occasion 1 is d. By preventively replacing an individual before its failure, we may need a new individual of the

corresponding type in the planning horizon. For example, individual $I_{2,5}$ is needed in Figure 1(b) but not in Figure 1(a). In addition to grouping PM with CM, we can also group PM with PM activities. At group maintenance occasion 2, individuals $I_{1,5}$ and $I_{2,3}$ are preventively replaced at the same time, which similarly reduces one setup cost. Notice that the group structure shown in Figure 1 (b) is just a possible option, not necessarily the optimal one.

Next, we begin the model development by first defining the planning horizon. Suppose the planning horizon is [0, S] and define time s as the current time when we need to make a decision. Assume that failure and maintenance only take place at some discretized times $\{s, s + \delta, \ldots, s + T\delta\}$, where δ is the length between two consecutive decision epochs and the number of decision epochs is $T = \left\lceil \frac{S-s}{\delta} \right\rceil$. For notational convenience, we will use [0, T] as the planning horizon for the remainder of this paper.

Let $\xi \in \mathsf{B}^n$ denote the vector of working individuals' states at the current time. If the individual of component i is functioning, $\xi_i = 0$, otherwise $\xi_i = 1$. At each maintenance decision epoch, an individual that needs be maintained is replaced with a new one of the same type. For any component type, the number of individuals needed for replacement in a given planning horizon is unknown at the current time, because it also depends on maintenance decisions. Let q denote the maximum number of individuals needed over the planning horizon [0, T]. This maximum is obtained when a replacement is performed at each decision epoch, $q = T + 1$. The first-stage decision variable is defined as follows.

Let

$$x_i = \begin{cases} 1, & \text{if the individual of component } i \text{ is replaced at the current time,} \quad i \in N \\ 0, & \text{otherwise,} \quad i \in N \end{cases}$$

The total replacement cost at the first stage consists of two parts: CR cost and PR cost. Since a failed individual at the current time ($\xi_i = 1$) needs to be correctively replaced, the total CR cost at the first stage is $\sum_{i\in N} c_{i,\mathrm{CR}}\xi_i$. To avoid introducing additional decision variables, the total PR cost at the first stage is expressed by a math equivalent $\sum_{i\in N} c_{i,\mathrm{PR}}x_i - \sum_{i\in N} c_{i,\mathrm{PR}}\xi_i$. Therefore, the total replacement cost at the first stage is $\sum_{i\in N} c_{i,\mathrm{PR}}x_i + \sum_{i\in N}\left(c_{i,\mathrm{CR}} - c_{i,\mathrm{PR}}\right)\xi_i$.

Our objective is to find the minimum expected total maintenance cost over the planning horizon [0, T].

$$\min\ dz + \sum_{i\in N} c_{i,\mathrm{PR}}x_i + \sum_{i\in N}\left(c_{i,\mathrm{CR}} - c_{i,\mathrm{PR}}\right)\xi_i + Q(\mathbf{x}),$$

subject to:

$$\begin{aligned} & x_i \geq \xi_i, && i \in N \\ & z \geq x_i, && i \in N \\ & x_i \in \{0,1\}, && i \in N \\ & z \in \{0,1\} \end{aligned}$$

where $Q(\mathbf{x}) = E_{\omega\in\Omega}\left(Q(\mathbf{x},\omega)\right)$ is the expected cost for the second-stage problem which includes all decision times after the current time s, and $Q(\mathbf{x}, \omega)$ is the second-stage cost for scenario ω. We defer the detailed derivation of $Q(\mathbf{x}, \omega)$ to the next section after presenting the DEF. Decision variable z indicates whether any maintenance action occurs at the current time.

The decision at each decision time concerns which individuals should be selected for PR. If an individual has failed at a decision time, it has to be correctively replaced, and this also presents an opportunity for other aged yet functioning individuals to be preventively replaced. We next present the DEF of the proposed two-stage stochastic programming problem. Additional decision variables are needed to develop the DEF model. These decision variables are defined as follows:

$$\tilde{x}_{it}^{r\omega} = \begin{cases} 1, \text{ if } I_{ir} \text{ is replaced at or before time } t \text{ in scenario } \omega, & i \in N, t \in T_{\mathrm{s}}, r \in R, \omega \in \Omega \\ 0, \text{ otherwise.} & i \in N, t \in T_{\mathrm{s}}, r \in R, \omega \in \Omega \end{cases}$$

and

$$z_t^{\omega} = \begin{cases} 1, \text{ if maintenance occurs at time } t \text{ in scenario } \omega, & t \in T_{\mathrm{s}}, \omega \in \Omega \\ 0, \text{ otherwise.} & t \in T_{\mathrm{s}}, \omega \in \Omega \end{cases}.$$

To facilitate the model development, we introduce two auxiliary variables $Y_i^{r\omega}$ and $w_{it}^{r\omega}$ based on $\tilde{x}_{it}^{r\omega}$. The variable $Y_i^{r\omega}$ is an indicator of the maintenance type. More details regarding these auxiliary variables will be discussed in section 2.2.1. The DEF of this problem can be described as follows.

Model DEF:

minimize

$$\sum_{\omega\in\Omega} p(\omega)\left(\sum_{i\in N}\left(\underbrace{\sum_{r=1}^{q}\left(c_{i,\mathrm{PR}}Y_i^{r\omega}\right)}_{C_{i,\mathrm{PR}}}+\underbrace{\sum_{r=1}^{q}\left(c_{i,\mathrm{CR}}\left(1-Y_i^{r\omega}\right)\right)-c_{i,\mathrm{CR}}\left(1-\tilde{x}_{iT}^{r\omega}\right)}_{C_{i,\mathrm{CR}}}\right)+\underbrace{\sum_{t\in T_{\mathrm{s}}} dz_t^{\omega}}_{C_{\mathrm{s}}}\right) \quad (1a)$$

subject to

$$\tilde{x}_{it}^{r\omega} \le \tilde{x}_{i,t+1}^{r\omega}, \qquad i\in N, t\in T_{\mathrm{s}}\setminus\{T\}, r\in R, \omega\in\Omega \quad (1b)$$

$$\tilde{x}_{i,t+1}^{r+1,\omega} \le \tilde{x}_{it}^{r\omega}, \qquad i\in N, t\in T_{\mathrm{s}}\setminus\{T\}, r\in R\setminus\{q\}, \omega\in\Omega \quad (1c)$$

$$\sum_{r\in R}\left(\tilde{x}_{it}^{r\omega}-\tilde{x}_{i,t-1}^{r\omega}\right)\le z_t^{\omega}, \qquad i\in N, t\in T_{\mathrm{s}}\setminus\{0\}, \omega\in\Omega \quad (1d)$$

$$\tilde{x}_{i0}^{1\omega} \le z_0^{\omega}, \qquad i\in N, \omega\in\Omega \quad (1e)$$

$$\tilde{x}_{it}^{r\omega} \le \tilde{x}_{i,t+T_{i,r+1}^{\omega}}^{r+1,\omega}, \qquad i\in N, t\in[0,\ T-T_{i,r+1}^{\omega}], r\in R\setminus\{q\}, \omega\in\Omega \quad (1f)$$

$$\tilde{x}_{iT_{i1}^{\omega}}^{1\omega} = 1, \qquad i\in\{j\in N \mid T_{j1}^{\omega}\le T\}, \omega\in\Omega \quad (1g)$$

$$\tilde{x}_{i0}^{r\omega} = 0, \qquad i\in N, r\in R\setminus\{1\}, \omega\in\Omega \quad (1h)$$

$$x_i = \tilde{x}_{i0}^{1\omega}, \qquad i\in N, \omega\in\Omega \quad (1i)$$

$$x_i \ge \xi_i, \qquad i\in N \quad (1j)$$

$$Y_i^{1\omega} = 1-w_{iT_{i1}^{\omega}}^{1\omega}, \qquad i\in N, \omega\in\Omega \quad (1k)$$

$$Y_i^{r\omega} = \left(\sum_{t=T_{ir}^{\omega}}^{T+T_{ir}^{\omega}}\left|y_{it}^{r\omega}\right|+\sum_{t=0}^{T_{ir}^{\omega}-1} w_{it}^{r\omega}\right)/2, \qquad i\in N, r\in R\setminus\{1\}, \omega\in\Omega \quad (1l)$$

$$y_{it}^{r\omega} = w_{it}^{r\omega} - w_{i,t-T_{ir}^{\omega}}^{r-1,\omega}, \qquad i \in N,\ r \in R \setminus \{1\},\ t \in [T_{ir}^{\omega},\ T'],\ \omega \in \Omega \tag{1m}$$

$$w_{it}^{r\omega} = \tilde{x}_{it}^{r\omega} - \tilde{x}_{i,t-1}^{r\omega}, \qquad i \in N,\ r \in R,\ t \in T_s \setminus \{0\},\ \omega \in \Omega \tag{1n}$$

$$w_{i0}^{r\omega} = \tilde{x}_{i0}^{r\omega}, \qquad i \in N,\ r \in R,\ \omega \in \Omega \tag{1o}$$

$$w_{it}^{r\omega} = 0, \qquad i \in N,\ r \in R,\ t \in [T+1,\ T'],\ \omega \in \Omega \tag{1p}$$

$$\tilde{x}_{i0}^{r\omega} \in \{0,1\}, \qquad i \in N,\ r \in R,\ t \in T_s,\ \omega \in \Omega \tag{1q}$$

$$x_i \in \{0,1\}, \qquad i \in N \tag{1r}$$

$$z_t^{\omega} \in \{0,1\}, \qquad t \in T_s,\ \omega \in \Omega \tag{1s}$$

$$w_{it}^{r\omega} \in \{0,1\}, \qquad i \in N,\ r \in R,\ t \in [0,\ T'],\ \omega \in \Omega \tag{1t}$$

$$Y_i^{r\omega} \in \{0,1\}, \qquad i \in N,\ r \in R,\ \omega \in \Omega \tag{1u}$$

Function (1a) is the objective function of the current problem. Decision variables x_i and z deal with maintenance decisions at the first-stage, and $\tilde{x}_{it}^{r\omega}$ and z_t^{ω} are the second-stage decisions for scenario ω. In the next section, we provide detailed derivation of the objective function.

2.2.1 Derivation of the Objective Function

In objective (1a), the total cost includes: (1) sum of the PR and CR costs incurred by individuals of component i in the planning horizon, denoted by $C_{i,\mathrm{PR}}$ and $C_{i,\mathrm{CR}}$ respectively, and (2) total system setup cost C_s. We break the derivation of the total cost function into the calculations of these cost elements.

- **Derivation of $C_{i,\mathrm{PR}}$**

For component i, the total cost of individuals preventively replaced over the planning horizon is given by

$$C_{i,PR} = \sum_{r=1}^{q} c_{i,\mathrm{PR}} Y_i^{r\omega} \tag{2}$$

where $Y_i^{r\omega}$ is defined in constraints (1k) and (1l).

We drop the superscript ω in the following discussions for notational convenience. From (1n), w_{it}^{r} determines when individual I_{ir} is replaced. If I_{ir} is replaced at time t, we have w_{it}^{r} = 1, and

$w_{it}^r = 0$ otherwise. The variable Y_i^r is used to determine the replacement type. This determination is a key element of the model development. If $Y_i^r = 1$, individual I_{ir} is preventively replaced, and $Y_i^r = 0$ implies that this individual is correctively replaced.

Next, we explain why Y_i^r can be used to identify the replacement type. It is obvious that the decision variables $\tilde{x}_{it}^{r\omega}$ and $w_{it}^{r\omega}$ only concern when a placement is performed, and have no indication on the type of replacement. For an individual I_{ir}, one way to determine its replacement type is to examine the time interval between the replacements of individuals $I_{i,r\text{-}1}$ and I_{ir}. Suppose that individuals $I_{i,r\text{-}1}$ and I_{ir} are replaced at times t_1 and t_2 (i.e., $w_{it_1}^{r-1} = 1$ and $w_{it_2}^{r} = 1$), respectively. If the difference between t_2 and t_1 equals to the lifetime of I_{ir}, namely T_{ir}, then I_{ir} is replaced at the end of its lifetime and the replacement type is CR. The replacement is PR otherwise.

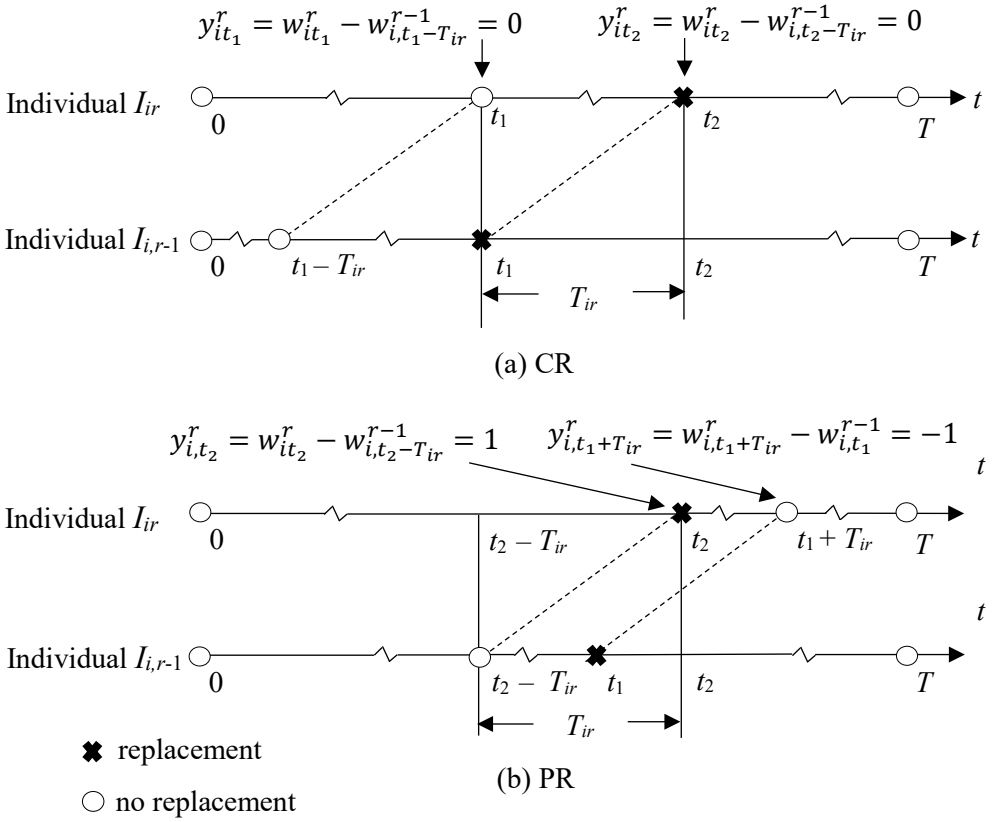

Figure 2: Illustration of distinguishing PR and CR

Therefore, if CR is performed on this individual, we have $w_{it}^{r} - w_{i,t-T_{ir}}^{r-1} = 0 \ \ \forall t \in [0,\ T]$, which leads to $\sum_{t=0}^{T} \left| y_{it}^{r} \right| = \sum_{t=0}^{T} \left| w_{it}^{r\omega} - w_{i,t-T_{ir}^{\omega}}^{r-1,\omega} \right| = 0$ (Figure 2(a)). If PR is performed on this individual, then $w_{i,t_2}^{r} - w_{i,t_2-T_{ir}}^{r-1} = 1$, $w_{i,t_1+T_{ir}}^{r} - w_{i,t_1}^{r-1} = -1$, and $w_{i,t+T_{ir}}^{r} - w_{it}^{r-1} = 0$ for all $t \in \{t \mid 0 \le t \le T, t \ne t_1, t \ne t_2\}$ (Figure 2(b)), and consequently, $\sum_{t=0}^{T} \left| y_{it}^{r} \right| = 2$. This makes the value of $\sum_{t=0}^{T} \left| y_{it}^{r} \right| / 2$ a good indicator for determining the replacement type, and $\sum_{t=0}^{T} \left| y_{it}^{r} \right|$ is calculated as follows:

$$\sum_{t=0}^{T} \left| y_{it}^{r} \right| = \sum_{t=0}^{T} \left| w_{it}^{r} - w_{i,t-T_{ir}}^{r-1} \right| \tag{3}$$

However, there are two boundary issues in Equation (3).

(1) Decision times of w_{it}^{r} need be extended beyond T. This is because Equation (3) does not count the maintenance decisions made for individual $I_{i,r\text{-}1}$ during the interval [$T - T_{ir} + 1$, T], which is due to no definition for the corresponding decisions for I_{ir} during the interval [$T + 1$, $T + T_{ir}$]. See the region labeled "Not Defined" in Figure 3 for an illustration. To include these decisions, the planning horizon for w_{it}^{r} is extended to $T' = T + \max_{i,r} T_{ir}$, and let $w_{it}^{r} = 0$ for $t > T$.

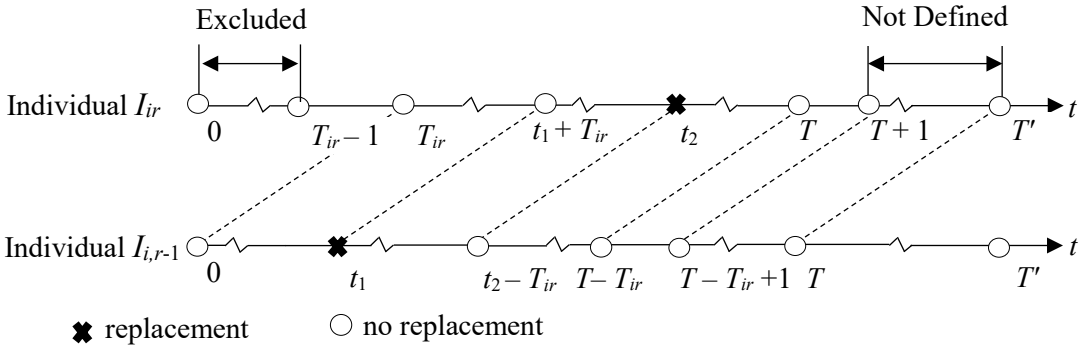

Figure 3: Illustration of the boundary issue of Equation (3)

(2) In Equation (3), the decision times considered for individual I_{ir} implicitly start from T_{ir}, and all decisions made before T_{ir} are excluded (illustrated in the region labeled "Excluded"). To recover decisions made for individual I_{ir} during the time interval [0, T_{ir} – 1], we add $\sum_{t=0}^{T_{ir}-1} w_{it}^{r}$ back to Equation (3). Equation (3) is now rewritten as follows,

$$Y_i^{r\omega} = \sum_{t=T_{ir}^{\omega}}^{T+T_{ir}^{\omega}} \left| y_{it}^{r\omega} \right| + \sum_{t=0}^{T_{ir}^{\omega}-1} w_{it}^{r\omega}, \qquad i \in N,\ r \in R \setminus \{1\},\ \omega \in \Omega .$$

The absolute function, $\left| y_{it}^{r\omega} \right|$, can be linearized by a pair of deviation variables $u_{it}^{r\omega}$ and $v_{it}^{r\omega}$ (Rardin and Rardin 2016). We replace $\left| y_{it}^{r\omega} \right|$ with Equation (4) in the constraint (1l), and add constraints (1v) to (1x) in the DEF model. Notice that constraint (1w) is not needed for linearization but it makes the problem formulation stronger when binary integer restrictions on $u_{it}^{r\omega}$ and $v_{it}^{r\omega}$ are relaxed.

$$\left| y_{it}^{r\omega} \right| = u_{it}^{r\omega} + v_{it}^{r\omega}, \qquad i \in N,\ r \in R,\ t \in [0, T'],\ \omega \in \Omega \tag{4}$$

$$y_{it}^{r\omega} = u_{it}^{r\omega} - v_{it}^{r\omega}, \qquad i \in N,\ r \in R,\ t \in [0, T'],\ \omega \in \Omega \tag{1v}$$

$$u_{it}^{r\omega} + v_{it}^{r\omega} \le 1, \qquad i \in N,\ r \in R,\ t \in [0, T'],\ \omega \in \Omega \tag{1w}$$

$$u_{it}^{r\omega}, v_{it}^{r\omega} \in \{0,1\}, \qquad i \in N,\ r \in R,\ t \in [0, T'],\ \omega \in \Omega \tag{1x}$$

- **Derivation of $C_{i,\mathrm{CR}}$ and C_{s}**

For component *i*, the total cost of individuals correctively replaced over the planning horizon is given by

$$C_{i,CR} = \sum_{r=1}^{q} \left(c_{i,\mathrm{CR}} \left(1 - Y_i^{r\omega} \right) - c_{i,\mathrm{CR}} \left(1 - \tilde{x}_{iT}^{r\omega} \right) \right) \tag{5}$$

As explained in the derivation of $C_{i,\mathrm{PR}}$, the expression $Y_i^r = 0$ implies a CR for individual I_{ir}. However, recall that for any component type, the number of individuals used for replacement is unknown due to the unknown maintenance decisions, and the maximum number of individuals needed for any component is considered in the optimization model. It is likely that some

individuals are not used in the planning horizon. If neither individual $I_{i,r\text{-}1}$ nor I_{ir} is used for replacement during the planning horizon, the value of Y_i^r is also zero. We need distinguish these two scenarios that both have $Y_i^r = 0$. This can be done by examining the value of $\tilde{x}_{iT}^{r\omega}$. If an individual is not used, we have $\tilde{x}_{iT}^{r\omega} = 0$ and $\tilde{x}_{iT}^{r\omega} = 1$ otherwise. The false corrective cost caused by an individual that is not used is $c_{i,CR}\left(1-\tilde{x}_{iT}^{r\omega}\right)$, and needs be subtracted from the total cost, $C_{i,\mathrm{CR}}$.

The last exception we need examine is when individual $I_{i,r\text{-}1}$ is replaced and individual I_{ir} is not in the planning horizon. In this situation, the value of Y_i^r is 0.5, which incurs the extra cost of $0.5(c_{i,\mathrm{PR}} - c_{i,\mathrm{CR}})$, according to Equations (2) and (5). To exclude the extra cost caused by this issue, we increase the maximum number of individuals needed, q, from $T + 1$ to $T + 2$. By doing so, it is guaranteed that at least one individual is not replaced during the planning horizon and this exception always occurs. Notice that this additional cost is a constant once we set $q = T + 2$ and does not affect the optimal maintenance decisions. Therefore, we do not subtract this additional cost from the total cost and can easily do so to obtain an accurate total expected cost. Lastly, the total setup cost over the planning horizon is $C_{\mathrm{s}} = \sum_{t\in T} dz_t^{\omega}$.

2.2.2 Constraints

Constraint (1b) is the definition of $\tilde{x}_{it}^{r\omega}$, which ensures that individual I_{ir} is replaced at or before $t + 1$ when it is replaced at or before t. Constraint (1c) implies that individual $I_{i,r+1}$ can only be replaced after I_{ir} is replaced. Constraints (1d) and (1e) ensure that the maintenance cost d incurs when any component is replaced at time t. Constraints (1f) and (1g) ensure that individual I_{ir} is replaced at the latest when it has been inside the system for T_{ir}^{ω} time units. In other words, I_{ir} has to be replaced before or at the end of its lifetime. Constraint (1h) implies that only

individual 1 could be replaced at time 0. In stochastic programming, it is required that the decision at $t = 0$ is the same as x_i for all scenarios, known as the non-anticipativity constraint, and this constraint is imposed by constraint (1i). The constraint (1j) forces all failed components at time $t = 0$ to be replaced. Constraints (1k) and (1l) define the auxiliary variable $Y_i^{r\omega}$, which is critical to identify the type of maintenance. Constraint (1m) provides the full definition of variable $y_{it}^{r\omega}$. Constraints (1n) – (1p) give the definition of variable $w_{it}^{r\omega}$. The remaining constraints (1q) – (1u) are binary constraints for all decision variables. The linearization of $\left|y_{it}^{r\omega}\right|$ can be found in Equation (4) and constraints (1v) – (1x).

3. Optimization algorithms

Our problem is a two-stage problem with pure binary decision variables. Properties of stochastic integer programs are scarce, and general efficient methods are lacking. We therefore design a heuristic algorithm under the framework of PHA to solve practical-size problems with up to 1,000 scenarios in moderate CPU time. To assess the performance of the proposed heuristic algorithm, we compare the performance of the proposed algorithms with three benchmark algorithms, namely, basic Benders decomposition (Algorithm 1), integer L-shaped method with Benders cuts (Algorithm 2) and standard PHA (Algorithm 3).

The basic Benders decomposition and integer L-shaped method with Benders cuts are considered as benchmark algorithms, since the LP relaxation and branch-and-cut approaches are basic ideas for solving integer programing problems. Standard PHA (Watson and Woodruff 2011) which decomposes a problem by scenarios instead of stages as in Benders decomposition provides a flexible framework for stochastic integer problem, and is also considered for comparison.

3.1 The Benchmark algorithms

3.1.1 Basic Benders decomposition algorithm

The basic Benders algorithm first solves the Benders master integer problem, and then solves the LP relaxation of subproblems to generate cuts which are added back to Benders master problem (Birge and Louveaux 2011). The procedure is repeated until no cuts found. We first define the initial master problem (MP) as follows:

$$\text{MP: min } dz + \sum_{i \in N} c_{i,\text{PR}} x_i + \sum_{i \in N} \left(c_{i,\text{CR}} - c_{i,\text{PR}} \right) \xi_i + \theta, \tag{6}$$

subject to:

$$\begin{aligned} &\theta \geq Q(\mathbf{x}), \\ &x_i \geq \xi_i, \qquad i \in N \\ &z \geq x_i, \qquad i \in N \\ &x_i \in \{0,1\}, \quad i \in N \\ &z \in \{0,1\} \end{aligned}$$

where $Q(\mathbf{x}) = \sum_{\omega \in \Omega} p(\omega) Q(\mathbf{x}, \omega)$ and $Q(\mathbf{x}, \omega)$ is the objective of the scenario ω in the second-stage problem, given by

$$Q(\mathbf{x}, \omega) = \sum_{i \in N} \left(C_{i,\text{PR}} + C_{i,\text{CR}} - c_{i,\text{PR}} x_i - \left(c_{i,\text{CR}} - c_{i,\text{PR}} \right) \xi_i \right) + C_{\text{s}} - dz \tag{7}$$

and subject to Constraints (1b) – (1x) except (1e), (1j) and (1r). Constraints(1e), (1j) and (1r) are excluded from the sub-problem since they only concern the decision variables in the first-stage. For each scenario ω, Benders cut can be written as

$$\theta_\omega \geq \mathbf{E}_{m,\omega} \mathbf{x} - e_{m,\omega}, \quad m \in \{1, \ldots, M\}, \ \omega \in \Omega, \tag{8}$$

where M denotes the maximum iterations.

To improve the performance of the basic Benders decomposition problem, we use the multi-cut strategy, since research has shown that a multi-cut strategy can lead to a faster convergence compared with single-cut (Birge and Louveaux 2011).

Algorithm 1: (Basic Benders decomposition)
1: **Initialization:** $\theta_\omega \leftarrow -\infty$, for $\forall\omega \in \Omega$, $\varepsilon \leftarrow 10^{-2}$, and assign an integer feasible **x** to the sub-problem.
2: Solve the LP relaxation of the sub-problem, $Q(\mathbf{x}, \omega)$, for each ω in Ω.
3: **If** $\theta_\omega - Q(\mathbf{x}, \omega) \leq \varepsilon$, $\forall\omega\in\Omega$, **return** optimal solution: $(\mathbf{x}^*, \theta_\omega^*) \leftarrow (\mathbf{x}, \theta_\omega)$. **Else**, go to step 4.
4: Add Benders cuts using Equation (8) into the MP, where $\theta = \sum_{\omega\in\Omega} p(\omega)\theta_\omega$.
5: Solve the MP as IP to get new θ_ω, $\forall\omega \in \Omega$. Go to step 2.

3.1.2. Integer L-shaped method with Benders cuts

In Algorithm 2, we initialize Benders master problem with Benders cuts. More specifically, the root note is obtained by solving the LP relaxation of the master problem via Benders decomposition and keeping the cuts. In the branch-and-cut process, at each node, if the solution is integer feasible, the subproblem is solved to generate integer optimality cuts which are defined as follows (Laporte and Louveaux 1993):

$$\theta \geq \left(Q\left(\mathbf{x}^*\right) - L\right)\left(\sum_{i\in S\left(\mathbf{x}^*\right)} x_i - \sum_{i\notin S\left(\mathbf{x}^*\right)} x_i - \left|S\left(\mathbf{x}^*\right)\right|\right) + Q\left(\mathbf{x}^*\right), \tag{9}$$

where $S\left(\mathbf{x}^*\right) := \left\{i \mid x_i^* = 1\right\}$.

In addition to the integer optimality cuts, Benders cuts are also generated and added if violated by the candidate solution into the MP, in order to improve the performance of the Integer L-shaped method. Therefore, for each node in the branch-and-cut search tree, if the candidate solution is integer feasible, both Benders cuts and integer optimality cuts are added via

lazy constraint callback routine, otherwise only Benders cuts are added by using user-cut callback routine.

Algorithm 2: (Integer L-shaped method with Benders cuts)

1: **Initialization** $\theta^* \leftarrow +\infty$;
Initialize the MP by solving the LP relaxation via Benders, and keep cuts $\Rightarrow (\mathbf{x}, \theta)$

2: **Branch and Cut**
At each node in the search tree:
Solve LP relaxation $\Rightarrow (\mathbf{x}, \theta)$
If LP bound exceeds known incumbent θ^*, prune.
If x is integer feasible:
Solve subproblem $Q(\mathbf{x})$ to generate integer optimality cuts using Equation (9).
Solve LP relaxation of the subproblem $Q(\mathbf{x})$ to generate Benders cuts.
If $(\mathbf{x}, \theta)$ violates any Benders cut or integer optimality cut, add cut to LP relaxation of the MP and resolve.
Else, update the incumbent, $\theta^* \leftarrow \theta$
If x is not integer feasible:
Solve LP relaxation of the subproblem $Q(\mathbf{x})$ to generate Benders cuts.
If $(\mathbf{x}, \theta)$ violates any Benders cut, add cut to LP relaxation and resolve.
Else, branch to create new nodes.

3.1.3. Standard progressive hedging algorithm

We also examine the performance of the standard PHA on our problem. PHA (Rockafellar and Wets 1991) is more memory-saving compared with solving the DEF directly because of the scenario-by-scenario decomposition. The penalty factor ρ can significantly affect the speed of convergence and the solution quality. The ρ selection is recommended to be close in the magnitude to the coefficient of first-stage variable (Watson and Woodruff 2011). In our computational study, ρ is set to 50. Details of the PHA are described in Algorithm 3. A different form of the objective function, $\mathbf{cx}+E(Q(\mathbf{x}, \omega))$, is used for concise presentation of the algorithm (Gade et al. 2016).

Algorithm 3: (The standard PHA)

1. **Initialization**:
   Let $v \leftarrow 0$, $\varepsilon \leftarrow 10^{-2}$;
   $\mathbf{x}_\omega^{(v)} \leftarrow \arg\min_{\mathbf{x}} (\mathbf{cx} + Q(\mathbf{x}, \omega))$, $\forall \omega \in \Omega$;
   $\bar{\mathbf{x}}^v \leftarrow \sum_{\omega \in \Omega} p(\omega) \mathbf{x}_\omega^v$;
   $\mathbf{w}_\omega^v \leftarrow \rho\left(\mathbf{x}_\omega^v - \bar{\mathbf{x}}^v\right)$, $\forall \omega \in \Omega$.
2. **Update iteration variable**: $v \leftarrow v + 1$.
3. **Decomposition**:
   $\mathbf{x}_\omega^{(v)} \leftarrow \arg\min_{\mathbf{x}} (\mathbf{cx} + \mathbf{w}_\omega^{v-1}\mathbf{x} + \frac{\rho}{2}\left\|\mathbf{x} - \bar{\mathbf{x}}^{v-1}\right\| + Q(\mathbf{x}, \omega))$, $\forall \omega \in \Omega$.
4. **Aggregation**: $\bar{\mathbf{x}}^v \leftarrow \sum_{\omega \in \Omega} p(\omega) \mathbf{x}_\omega^v$.
5. **Update price**: $\mathbf{w}_\omega^v \leftarrow \mathbf{w}_\omega^{v-1} + \rho\left(\mathbf{x}_\omega^v - \bar{\mathbf{x}}^v\right)$, $\forall \omega \in \Omega$.
6. **Calculate converge distance**: $g^v \leftarrow \sum_{\omega \in \Omega} p(\omega)\left\|\mathbf{x}_\omega^v - \bar{\mathbf{x}}^v\right\|$, $\forall \omega \in \Omega$.
7. **Termination**: **If** $g^v < \varepsilon$, stop and **return** optimal solution $\bar{\mathbf{x}}^v$. **Else**, go to step 2.

3.2. Progressive-hedging-based Heuristic Algorithm

Algorithm 1 cannot provide meaningful results due to the LP relaxation employed, and becomes more difficult and time-consuming as more cuts are added. Algorithm 2 is also computational intensive as the number of binary variables and constraints increases. Standard PHA similarly suffers the computational intractability, since even for a small-scale multi-component maintenance problem, the scenario sub-problem in the DEF can have a large number of decision variables and constraints, beyond what commercial solvers (e.g., CPLEX) can handle. However, PHA provides flexible framework for solving stochastic integer problems. To address the bottleneck in solving the scenario sub-problem using the standard PHA, we develop an efficient heuristic algorithm based on the problem structure for the scenario sub-problem. Before describing the details of the heuristic, we first present two properties regarding the grouping, which the proposed algorithm heavily relies on.

Theorem 1. Let ε_m denote the minimum not-used-residual lifetime of all individuals in group m in a group structure W_T in the planning horizon [0, T], and m_{last} represent the last group in W_T. There exists an optimal grouping structure W_T^* such that $\varepsilon_m \leq 1, \forall\ m \in W_T^* \setminus m_{\text{last}}$. (Proof is in Appendix A.1).

Theorem 2. Given a set of working individuals sorted according to their failure times, there exists an optimal grouping structure W^* for this set such that maintenance activities are executed at the same consecutive order. (Proof is shown in Appendix A.2).

Theorem 1 helps determine tentative replacement schedules for each individual. Based on Theorem 1, we select two tentative replacement schedules for individuals without considering economic dependence, replacing one time unit before a failure or at the failure. Theorem 2 further ensures that it is optimal to execute the replacement activities for all working individuals in the order as tentatively planned. This significantly decreases the number of possible grouping structure options needed to be considered in the heuristic, and thus substantially reduce the algorithm complexity.

The basic idea of the heuristic algorithm is as follows. Given a scenario sub-problem, the heuristic iteratively identifies the optimal group structure for the *working* individuals. At each iteration it first obtains tentative replacement schedules for all *working* individuals based on Theorem 1. It then considers a shifting window (ι), which is also a decision variable. The tentative replacement time of an individual can only be shifted to the left of the time axis within the given time window for grouping. The goal of the heuristic is to reduce the setup costs by grouping maintenance activities, and ultimately reduce the total maintenance cost. The time complexity of the heuristic algorithm is polynomial (see Appendix A.3 for proof).

We next introduce some important definitions the proposed heuristic will make significant use of. Let K denote the set of all *working* individuals at the current iteration of the heuristic. Let τ_{ij} and τ'_{ij} denote the tentative and definitive replacement times of individual I_{ij}, respectively. Let K' represent the sorted set of K according to their tentative replace times. For example, consider a four-component system, and the tentative replacement times of four individuals at one iteration are provided in Table 1. In this example, we have $K = \{I_{1,5}, I_{2,3}, I_{3,2}, I_{4,4}\}$ and $K' = \{I_{2,3}, I_{1,5}, I_{4,4}, I_{3,2}\}$, as illustrated in Figure 4.

Table 1: Example replacement information for four components

working individual	$I_{2,3}$	$I_{1,5}$	$I_{4,4}$	$I_{3,2}$
tentative replacement time ($\tau_{i,j}$)	16	18	20	21

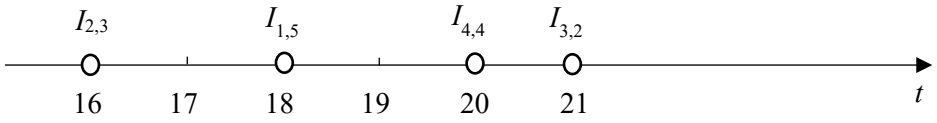

Figure 4: Working individuals at one iteration

At any iteration, the heuristic seeks the optimal *group structure* for all individuals in set K, using the **Grouping Rule**. The *group structure* is a collection of groups such that all individuals within each group are jointly replaced by their immediate descendent individuals at the same time. Let set W represent the optimal group structure at the current iteration. The individuals in any group within set W is ordered based on their tentative replacement schedules, and the groups in W are ordered based on their definitive replacement times. For example, $W = \{\{I_{2,3}\}, \{I_{4,4}, I_{3,2}\}\}$ indicates that the optimal group structure contains two groups, $\{I_{2,3}\}$ and $\{I_{4,4}, I_{3,2}\}$, and group $\{I_{2,3}\}$ is replaced before group $\{I_{4,4}, I_{3,2}\}$ (illustrated in Figure 5). We can also see from Figure 5 that the tentative replacement time of the first individual in any group in set W becomes the definitive replacement time for all individuals in the corresponding group. In the illustrative

example, tentative replacement time of $I_{4,4}$ becomes the definitive replacement times of $I_{4,4}$ and $I_{3,2}$. The optimal group structure over the entire planning horizon can be obtained as the union of Ws at all iterations, denoted as W_T.

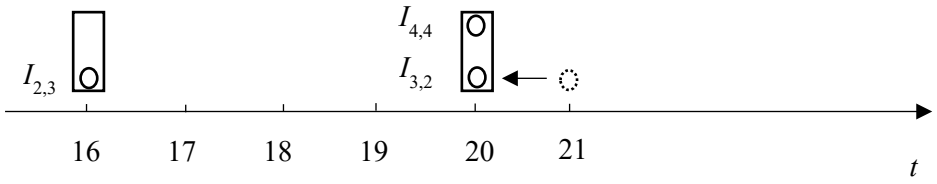

Figure 5: A group structure at one iteration

It is possible that set W does not contain all individuals in set K at an iteration. For instance, in previous example, $I_{1,5}$ is included in set K but not in any group in set W. This is because a system is often composed of different individuals with different life time cycles. For example, the tentative replacement times for the working pump, valve and bearings in a chemical plant may be two years, four months and six months from now, respectively, and it is not economic to replace the pump with the valve or bearing at the same time.

At each iteration, if an individual in set K is included in some group in set W, it means that it is replaced by its corresponding new individual. If this new individual has a tentative replacement schedule beyond the planning horizon, it is removed from set K, since no additional individual from this component type is needed in the planning horizon. Individual(s) that is not in set W but remains in set K is considered for grouping with the other newly replaced individuals at the next iteration. In the previous example, individuals $I_{2,3}$, $I_{4,4}$ and $I_{3,2}$ are replaced by their descendants $I_{2,4}$, $I_{4,5}$ and $I_{3,3}$, respectively, and $I_{1,5}$ is not grouped with any other individual, so set K at the next iteration is $\{I_{1,5}, I_{2,4}, I_{3,3}, I_{4,5}\}$, as shown in Figure 6. The heuristic stops when set K is empty.

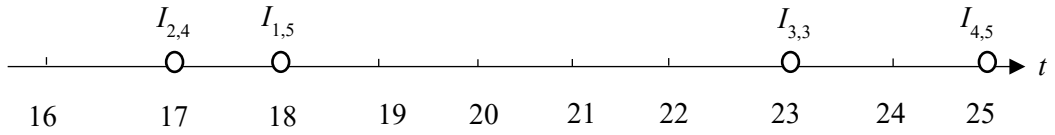

Figure 6: Working individuals at the next iteration

The **Grouping Rule** performed on all individuals in set K works as follows. Let $K'[i]$ represent the individual in the i^{th} position in set K'. We compare several group structures and select the one that gives the minimum weighted PHA replacement cost cumulated till the current iteration as the optimal group structure. The weighted PHA replacement cost is calculated according to step 3 in Algorithm 3. A simple scheme is used in the **Grouping Rule** to generate a set of candidate group structures. Specifically, group structure m starts with $K'[m]$. Any individual before $K'[m]$ is not considered for grouping in group structure m. The total number of group structures generated using this method is $|K'| - 1$.

Suppose the tentative replacement time of $K'[m]$ is t. We then group all individuals with tentative replacement times before or at $t + \iota$ if the definitive replacement time of the predecessor of the current working individual is before time t. The individuals that can be shifted are simultaneously replaced at time t. As we have shown in Figure 5, time t is now the definitive replacement times of all these individuals in this group. Let $K'[\upsilon]$ represent the last individual grouped with $K'[m]$. We next start with $K'[\upsilon+1]$ and identify individuals that are after $K'[\upsilon+1]$ and can be grouped with $K'[\upsilon+1]$. The grouping process terminates when $\upsilon = |K'|$, meaning no more individual can be grouped. If individual $K'[1]$ is not grouped with other individual(s) during the grouping process, it will be made a one-individual group in W in order to keep the grouping process rolling in the horizon.

We use the same four-component system considered earlier to illustrate the grouping process in the **Grouping Rule**. Suppose the shifting window $\iota = 3$. The three candidate group structures are illustrated in Figure 7.

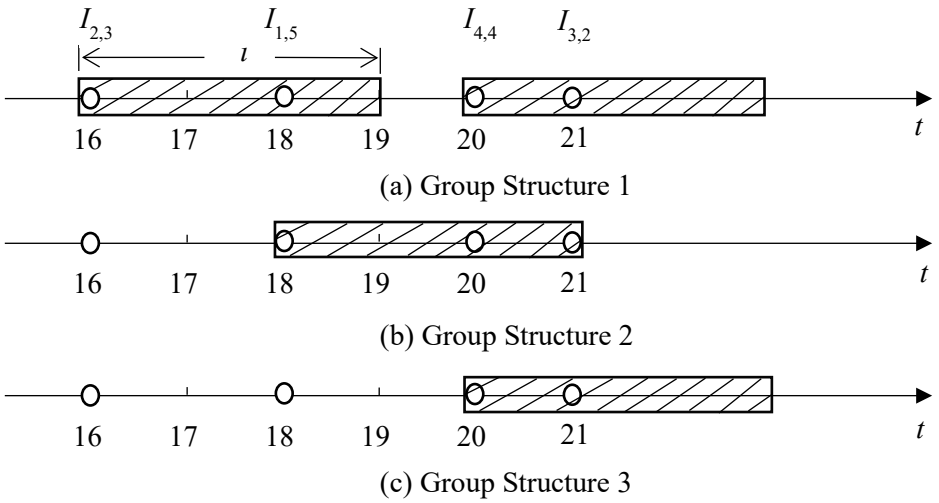

Figure 7: Group structures at one iteration

In group structure 1, the grouping starts with $K'[1]$ which is $I_{2,3}$. Following the rule, there are two groups in group structure 1 (Figure 7(a)), $\{I_{2,3}, I_{1,5}\}$ and $\{I_{4,4}, I_{3,2}\}$. In the second group structure, the grouping starts from $K'[2]$ which is $I_{1,5}$, and three individuals, $I_{1,5}$, $I_{4,4}$, and $I_{3,2}$ are grouped (Figure 7(b)). In the last group structure, we start with $K'[3]$, and group $I_{4,4}$ and $I_{3,2}$. Notice that $I_{2,3}$ is forcefully replaced in group structures 2 and 3 for the purpose of moving the grouping process in the horizon. Among all three options, group structure 3 gives the minimum weighted PHA cost cumulated, and therefore $W = \{\{I_{2,3}\}, \{I_{4,4}, I_{3,2}\}\}$.

We further define some additional variables to facilitate the description of the heuristic algorithm. Let Δ denote the not-used-residual lifetime of an individual, e.g., $\Delta = 1$ meaning that an individual has one lifetime unit not used when it is replaced. Let $T(K'[i])$, $\tau(K'[i])$ and $\tau'(K'[i])$

represent the lifetime, tentative and definitive replace times of individual $K'[i]$, respectively. Details of the heuristic are in Algorithm 4 and the Grouping Rule.

Algorithm 4: (Heuristic algorithm for one scenario)

Initialization: $\Delta \leftarrow \{0, 1\}$, and determine a set of values for ι, $\iota = \{\iota_1, \iota_2, \ldots\}$
For all combinations of Δ and ι, select the one that gives the minimum total weighted PHA replacement cost and return the corresponding optimal group structure W

1: **Initialization**: Assign tentative replacement times for the first individual of each component
$K \leftarrow \{I_{1,1}, I_{2,1}, \ldots, I_{n,1}\}$, and $\tau_{i,1} \leftarrow T_{i,1} - \Delta$, $\forall\, i \in N$
2: Apply **Grouping Rule** to obtain the optimal group structure W.
3: Update Set K.
$\forall\, I_{ij} \in W$
Replace I_{ij} in set K with $I_{i,j+1}$
Assign tentative replacement schedule to $I_{i,j+1}$, $\tau_{i,j+1} \leftarrow \tau'_{ij} + (T_{i,j+1} - \Delta)$
$\forall\, I_{ij} \in K$, **If** $\tau_{ij} > T$, remove I_{ij} from set K.
If K is empty, stop. **Else**, go to step 2.

Grouping Rule

1: Sort K in ascending order based on $\tau_{ij} \Rightarrow$ sorted set K'

2: Select the structure m that gives the lowest weighted PHA replacement cost cumulated $\Rightarrow W$.
Group Structure *m*: m from 1 to $|K'| - 1$
2.1: Initialize the last individual grouped as $K'[\upsilon]$: $\upsilon \leftarrow m$
2.2: $t \leftarrow \tau(K'[\upsilon])$
Group individuals in set K' if the definitive replacement times of their predecessors are before t until $\tau(K'[\upsilon']) > t + \iota$, $\upsilon' = \upsilon + 1, \upsilon + 2, \ldots$
2.3: Let ϑ denote the position of the last individual grouped in step 2.2
Update definitive replacement times: $\tau'(K'[\upsilon']) \leftarrow t$, $\upsilon' = \upsilon + 1, \upsilon + 2, \ldots, \vartheta$
$\upsilon \leftarrow \vartheta + 1$
2.4: **If** $\upsilon \geq |K'|$, compute the weighted PHA replacement cost cumulated, then stop. **Else**, go to step 2.2.

3: **If** $K'[1] \notin W$: $W \leftarrow W \cup \{ (K'[1]) \}$

4: **Return** set W.

4. Computational study

We perform our computational study on a computer with a CPU of Intel i7-6700, 3.4G Hz and a RAM of 16G. A python based package Pyomo (Hart et al. 2011, Hart et al. 2012) is used to implement the algorithms with the solver of CPLEX v12.7.1.

For computational tests, we assume that all components' lifetimes follow Weibull distributions. For each component, we draw the shape and scale parameters from uniform distributions $U(4, 7)$ and $U(1, 8)$, respectively. Without loss of generality, the cost of PR ($c_{i,\mathrm{PR}}$) is assumed to be 1. The cost of CR ($c_{i,\mathrm{CR}}$) is drawn from a uniform distribution $U(6, 16)$. Suppose that setup cost d is 5 and the current time s is 2. Assume that the individual of component 1 is failed at the current time. Table 3 summarizes the parameters considered in this computational study.

Table 3: Component parameters

i	shape	scale	$c_{i,\mathrm{CR}}$	rounded MTTF
1	5.7	7.3	11.4	7
2	6.7	4.2	9.1	4
3	5.6	3.2	8	3
4	5.4	2.9	11.1	3
5	5.5	2	14.2	2
6	4.9	4.6	7.4	4
7	6.7	6.4	9.8	6
8	4.8	1.9	14.6	2

The problem size is mainly determined by three factors, the number of components, the length of the planning horizon, and the number of scenarios. The number of constraints is instance-dependent because of constraints (1f) and (1m), and we approximate it with the maximum possible number of constraints. Table 4 shows the problem size for some combinations of n, T and $|\Omega|$. From Table 4, we can see that even for a small problem ($n = 8$, $T = 40$, $|\Omega| = 100$), the total number of decision variables and constraints are more than 9 million and 27 million, respectively. The problem size grows exponentially when n, T, and/or $|\Omega|$ increase.

Table 4: Illustration of problem size

case	n	T	$\lvert\Omega\rvert$	Variables		Constraints	
				master	sub-problem	master	sub-

				problem		problem	problem
1	4	10	20	5	158,600	88	430,220
2	4	20	20	5	361,200	88	1,002,420
3	6	30	50	7	2,353,500	312	6,624,050
4	6	30	100	7	4,707,000	612	13,248,100
5	8	30	100	9	6,275,000	816	17,663,100
6	8	40	100	9	9,580,000	816	27,224,100

We compare the performance of the three benchmark algorithms with the proposed heuristic algorithm. For standard PHA, we run our experiments in a stochastic programming package PySP inside Pyomo (Hart et al. 2011, Hart et al. 2012).

Table 5 summarizes the performance of all four algorithms. Objective values of DEF are also provided, and they are the true objective values. If out of memory is encountered or computation time is longer than 1 day, NA is reported. Algorithm 1 (basic Benders decomposition) is faster than Algorithm 2 (Integer L-shaped methods with Benders cuts) and less memory-consuming than Algorithm 2. For example, Algorithm 1 is capable of solving cases 5 and 6, but Algorithm 2 cannot. However, Algorithm 1 solves the LP relaxation of sub-problems, and the decisions are therefore non-integer valued. This leads to negative objective values that are meaningless. By employing integer optimality cuts, which is less efficient in computing, Algorithm 2 provides meaningful objective results. Algorithm 3 also gives the true objective value for all test cases considered. Compared with Algorithm 2, Algorithm 3 converges faster, consumes less memory and has a better solution quality.

However, Algorithms 1 – 3 cannot solve large-scale problems, e.g., $|\Omega| = 1000$. From Table 4, we can see that for cases 25 to 36, Algorithm 4 is the only algorithm that can solve the problem within a reasonable amount of time. On average, Algorithm 4 is about six times faster than Algorithm 3 and 13 times faster than Algorithm 1 in our computational studies.

We further examine the performance of Algorithm 4. We compute the percentage error by comparing the resulted costs from Algorithm 4 with the true objective values of DEF models. From Table 5, we can see that the percentage error decreases as the problem size increases. For example, the percentage error is 13% in case 1 where there are four components and drops to 2% in case 10 where the system has ten components. This is because more grouping activities are available at each iteration in the heuristic algorithm as the number of components increases, and therefore the performance of the heuristic approaches that of the DEF which gives the true optimal decision. Based on our computational studies, the proposed heuristic algorithm (Algorithm 4) performs well in terms of both efficiency and effectiveness for practically large-scale problems.

Table 5: Algorithm performance (part 1, $|\Omega| = 50$)

			DEF		Algorithm 1				Algorithm 2			Algorithm 3			Algorithm 4			
case	n	T	CPU time (sec.)	Obj.	Iter.	Cuts	CPU time (sec.)	Obj.	Cuts	CPU time (sec.)	Obj.	Iter.	CPU time (sec.)	Obj.	Iter.	CPU time (sec.)	Obj.	Obj. error %
1		10	119	89.33	8	342	214	-209.2	16	603	89.33	6	224	89.33	2	5	100.69	12.72%
2	4	20	NA		9	396	630	-402.8	17	2458	167.05	3	804	167.05	3	21	184.27	NA
3		30	NA		9	399	1247	-603.8	17	7049	242.35	6	4881	242.35	2	27	265.55	NA
4		10	77	194	17	700	664	-279.6	88	4610	194	13	530	194	2	21	205.04	5.69%
5	6	20	NA		20	881	2081	-530.9		NA		17	2900	375.22	3	94	390.03	NA
6		30	NA		22	989	4472	-792.1		NA		5	3823	552.09	3	184	569.73	NA
7		10	90	306.04	26	1080	1364	-385	364	25527	306.18	12	591	306.18	5	136	316.74	3.50%
8	8	20	NA		35	1508	4783	-721.3		NA		10	1984	597.71	5	402	612.49	NA
9		30	NA				NA			NA			NA		5	801	899.14	NA
10		10	68	525.46	50	2289	3231	-415.1		NA		10	632	525.62	5	348	535.92	1.99%
11	10	20	NA		50	2353	8689	-768.45		NA		10	2165	1033.9	5	1003	1046.1	NA
12		30	NA				NA			NA			NA		5	1941	1545.1	NA

Table 5 (cont'd): Algorithm performance (part 2, $|\Omega| = 100$)

			DEF		Algorithm 1				Algorithm 2			Algorithm 3			Algorithm 4			
case	n	T	CPU time (sec.)	Obj.	Iter.	Cuts	CPU time (sec.)	Obj.	Cuts	CPU time (sec.)	Obj.	Iter.	CPU time (sec.)	Obj.	Iter.	CPU time (sec.)	Obj.	Obj. error %
13		10	404	85.55	8	666	428	-212.4	17	1445	85.55	6	458	85.55	2	10	96.66	12.99%
14	4	20	NA		9	795	1269	-409.9	16	4529	159.7	3	1639	159.7	2	27	177.48	NA
15		30	NA		9	796	2528	-612.6					NA		2	53	258.6	NA
16		10	167	186.79	15	1236	1162	-285.4				13	1068	186.79	3	64	198.48	6.26%
17	6	20	NA		20	1742	4254	-543.5				16	5705	362.41	3	184	377.73	NA
18		30	NA				NA						NA		3	378	557.38	NA
19		10	187	295.38	26	2172	2659	-391.6				11	1054	295.41	5	271	306.31	3.70%
20	8	20	NA				NA			NA		11	4369	575.54	5	788	590.68	NA
21		30	NA				NA						NA		5	1569	875.94	NA
22		10	147	510.58	48	4324	6321	-423.74				11	1279	510.63	5	658	521.02	2.04%
23	10	20	NA				NA					11	6021	1005.6	5	1952	1017.8	NA
24		30	NA				NA						NA		5	3864	1516.4	NA

Table 5 (cont'd): Algorithm performance (part 3, $|\Omega| = 1000$)

			DEF		Algorithm 1			Algorithm 2		Algorithm 3		Algorithm 4			
case	n	T	CPU time (sec.)	Obj.	Iter.	Cuts	CPU time (sec.)	Obj.	Cuts	CPU time (sec.)	Obj.	Iter.	CPU time (sec.)	Obj.	Iter.	CPU time (sec.)	Obj.	Obj. error %
25		10										3	163	97.13				
26	4	20										3	436	177.05				
27		30										3	833	258.04				
28		10										3	659	200.34				
29	6	20										3	1899	380.27				
30		30	NA		NA			NA		NA		3	3711	559.9	NA			
31		10										5	2789	310.65				
32	8	20										5	8202	598.64				
33		30										5	16065	886.25				
34		10										5	6540	526.37				
35	10	20										5	19355	1027.8				
36		30										5	38880	1529.1				

5. Sensitivity analysis in the rolling horizon

In this section, we compare the proposed stochastic programming approach with a direct-grouping approach (Wildeman et al. 1997). The direct-grouping model uses a dynamic-programming algorithm that first finds the optimal replacement schedule for each component without considering economic dependence and then sort the components based on that. At iteration j, the algorithm identifies two groups that cover all maintenance activities of components 1 to j and provide the best savings for these components. The best grouping structure can be found by backtracking. This algorithm has in the worst case a time complexity of $o(n^2)$. However, the limitation of this algorithm is that it only considers the group structure of two groups at each iteration and ignores all other options (e.g., partition all maintenance activities into three or more groups).

We conduct a sensitivity analysis to examine the benefits of using the stochastic programming approach and the proposed heuristic algorithm. We consider a system with 10 components in a planning horizon [0, 20], and the length of decision period equals to 1. At each decision epoch, we consider a two-stage stochastic maintenance optimization problem where the second stage combines decisions of the remaining periods and each second stage has 1,000 scenarios. Finding the optimal policy over the planning horizon thus involves solving 20 two-stage problems. We repeat this procedure five times to obtain the average total maintenance cost over the planning horizon. The PR cost is assumed to be 1 for each type of component, and the CR costs are drawn from two different uniform distributions, $U(6, 16)$ and $U(17, 27)$. The lifetime of each individual is assumed to follow a Weibull distribution. We first draw the shape parameter from a uniform distribution $U(4, 7)$. To introduce more heterogeneity to the system, the scale parameter of the Weibull distributions is drawn from two different distributions, $U(1, 8)$

and $U(9, 20)$. Two levels of setup costs are considered. Details of the distributions and parameter values used in the sensitivity analysis are provided in Tables 6 and 7.

Table 6: Different levels of parameter

Level	Weibull scale parameter	d	cost of CR
High	$U(9, 20)$	100	$U(17, 27)$
Low	$U(1, 8)$	5	$U(6, 16)$

Table 7: Parameters for each type of component in different level

i	shape parameter	scale parameter		rounded MTTF		$c_{i,\mathrm{CR}}$	
		High	Low	High	Low	High	Low
1	5.7	18.5	9.2	17	9	18.4	11.4
2	6.7	11.3	5.7	11	5	16.1	9.1
3	5.6	8.9	4.5	8	4	15	8
4	5.4	8.3	4.2	8	4	18.1	11.1
5	5.5	6.3	3.2	6	3	21.2	14.2
6	4.9	12.2	6.1	11	6	14.4	7.4
7	6.7	16.3	8.2	15	8	16.8	9.8
8	4.8	6	3	5	3	21.6	14.6
9	4.6	6.3	3.2	6	3	21.5	14.5
10	4.4	4.7	2.4	4	2	19	12

Table 8 summarizes the comparison results. From Table 8, our approach clearly outperforms the benchmark one in all cases examined. In particular, the cost savings from the stochastic programming approach is much larger when the setup cost d is higher. We further notice that the saving is much more significant when the Weibull scale parameter is small. When the value of the scale parameter is smaller, components' lifetimes are shorter, and there will be more failures if PM is not effectively scheduled. This shows that our approach performs well when the setup cost is high and/or there are more maintenance activities that can potentially be grouped.

Table 8: Numerical example for rolling horizon comparison

case	Weibull scale parameter	*d*	cost of CR	Benchmark obj.	Algorithm 4 obj.	Savings
1	H	H	H	435	307.64	127.36
2	H	H	L	332.4	313.46	18.94
3	H	L	H	55	48.48	6.52
4	H	L	L	47.4	40.48	6.92
5	L	H	H	2613.34	1763.7	849.64
6	L	H	L	2356.34	1336.52	1019.82
7	L	L	H	788.06	284.38	503.68
8	L	L	L	526.06	267.86	258.2

6. Conclusion and future research

In this paper, we consider the problem of multi-component maintenance optimization over the finite planning horizon. By developing a set of novel modeling techniques, we use a stochastic programming approach to formulate the problem under realistic assumptions and design an efficient heuristic algorithm under the framework of the PHA for practical-size problems.

We compare our heuristic algorithm with three standard stochastic (mixed-) integer programming optimization algorithms. Our computation experiments show that the proposed algorithm in this study is the only one that are capable of handling practical-size problems of our interests. Furthermore, we compare the performance of the formulated two-stage stochastic maintenance model and the corresponding heuristic with that of a direct-grouping model using a dynamic-programing algorithm in the literature. The sensitivity analysis shows that significant savings can be obtained from the proposed stochastic programming approach.

Our work has extended the available literature in multi-component maintenance by proposing a novel and efficient stochastic programming approach. The modeling and solution techniques developed in this paper opens new research and implementation opportunities. Future research

will consider a different widely used maintenance policy, condition-based maintenance (CBM). CBM leverages sensor information on components' health status through inspection or real-time monitoring and aims to perform maintenance just in time by setting optimal control thresholds. Capturing these complexities requires a different problem formulation and different optimization algorithms. Moreover, maintenance activities are often subject to a pre-determined budget with a requirement on a system's reliability or availability. Future work will incorporate these constraints into the decision model. Lastly, it is worth extending the problem for more complex systems with stochastic and structural dependences, in addition to the economic dependence.

Appendix

(A.1) Theorem 1. Let ε_m denote the minimum not-used-residual lifetime of all individuals in group m in a group structure W_T in the planning horizon [0, T], and m_{last} represent the last group in W_T. There exists an optimal grouping structure W_T^* such that $\varepsilon_m \leq 1, \forall\, m \in W_T^* \setminus m_{\text{last}}$.

Proof:

We prove the theorem by contradiction. Suppose that there exists at least one group satisfying $\varepsilon_m > 1$ in any optimal group structure W_T^*. The goal is to show that by appropriately regrouping the maintenance activities, we can find an alternative group structure W_T' that yields no higher cost and has $\varepsilon_m \leq 1,\ \forall m \in W_T' \setminus m_{\text{last}}$.

We start the proof by first considering the case where there is only one group, say group λ, with $\varepsilon_\lambda > 1$ and $\varepsilon_m \leq 1,\ \forall\ m \in W_T' \setminus \{\lambda\}$. If there are more than one groups of this kind, we will start with the last group with $\varepsilon_m > 1$ and perform the regrouping process iteratively until $\varepsilon_m \leq 1$ for all groups.

Suppose groups λ and $\lambda + 1$ are replaced at t_λ and $t_{\lambda+1}$ respectively, and group λ is not the last group in group structure W_T^*. Next, we describe the details of how we construct the alternative group structure W_T'. We use the auxiliary variable w_{it}^r to help building the new group structure. We construct the new group structure by shifting and regrouping individuals in group λ and the subsequent groups when needed. There are three possible scenarios in the regrouping process.

Scenario 1: $t_{\lambda+1} \geq t_\lambda + \varepsilon_\lambda$.

In this scenario, we let all individuals in group λ replaced at $t_\lambda + \varepsilon_\lambda - 1$. For all individuals in group λ, we have $\left(w_{it}^r\right)'$ given by

$$\left(w_{it}^r\right)' = \begin{cases} w_{it}^r, & I_{ir} \notin \lambda, t \in T_s \\ 1, & I_{ir} \in \lambda, t = t_\lambda + \varepsilon_\lambda - 1 \\ 0, & I_{ir} \in \lambda, t \neq t_\lambda + \varepsilon_\lambda - 1 \end{cases}. \tag{A1}$$

Based on $\tilde{x}_{it'}^r = \sum_{t=0}^{t'} w_{it}^r$, we have,

$$\left(\tilde{x}_{it}^r\right)' = \begin{cases} \tilde{x}_{it}^r, & I_{ir} \notin \lambda, t \in T_s \\ 0, & I_{ir} \in \lambda, t < t_\lambda + \varepsilon_\lambda - 1 \\ 1, & I_{ir} \in \lambda, t \geq t_\lambda + \varepsilon_\lambda - 1 \end{cases}. \tag{A2}$$

From the definition of z_t, we have

$$\left(z_t\right)' = \begin{cases} z_t, & t \neq t_\lambda, t \neq t_\lambda + \varepsilon_\lambda - 1 \\ 0, & t = t_\lambda \\ 1, & t = t_\lambda + \varepsilon_\lambda - 1 \end{cases}. \tag{A3}$$

With the Equations (A1) – (A3), it is straightforward that solution $\left(w_{it}^r\right)'$ does not violate any constraint. It can also be easily verified that the cost remains the same in the reconstructed group structure under this scenario. The new replacement schedule of group λ make $\varepsilon_\lambda \leq 1$, but it may

cause some subsequent group(s) to have the minimum not-used-residual lifetime of all individuals in the group greater than one. We will perform this regrouping process recursively until all groups satisfy $\varepsilon_m \leq 1$.

Scenario 2: $t_{\lambda+1} < t_\lambda + \varepsilon_\lambda$.

Let ε_{ir} denote the not-used-residual life of individual I_{ir}. Define set P such that $\forall I_{ir} \in P$, we have $I_{ir} \in \lambda+1, I_{i,r-1} \notin \lambda$ and $\varepsilon_{ir} \leq 1$.We further separate Scenario 2 into two sub-scenarios based on whether set P is empty.

Scenario 2.1: $P \neq \varnothing$.

Define set Q such that $\forall I_{ir} \in Q$, we have $I_{ir} \in \lambda+1, I_{i,r-1} \in \lambda$. We construct two new groups λ' and $(\lambda + 1)'$, such that $\lambda' = \lambda \cup (\lambda+1) - Q$ and $(\lambda+1)' = Q$.

We replace all individuals in group λ' at time $t_{\lambda+1}$. It is obvious that the new group λ' satisfies $\varepsilon_{\lambda'} \leq 1$. We then process group $(\lambda + 1)'$ in the same way as how we process group λ from the beginning. It is also obvious that no additional cost incurs during this regrouping. And we will regroup recursively until all groups satisfy $\varepsilon_m \leq 1$.

If set Q is empty, we actually combined two groups into one group and satisfy $\varepsilon_m \leq 1$for all groups while saving one setup cost.

We can similarly show that no constraint is violated during the regrouping process in this scenario, and detailed proof is omitted.

Scenario 2.2: $P = \varnothing$

We construct the same two new groups λ' and $(\lambda + 1)'$ as in Scenario 2.1. The difference is that the new group λ' in this scenario does not satisfy $\varepsilon_{\lambda'} \leq 1$at replacement time $t_{\lambda+1}$. We next

process group λ' in the same way as how we process group λ from the beginning and then do the same thing for group $(\lambda + 1)'$.

Proof completed.

(A.2) Theorem 2. Given a set of working individuals sorted according to their failure times, there exists an optimal grouping structure W^* for this set such that maintenance activities are executed at the same consecutive order.

Proof:

We prove Theorem 2 by contraction.

For two individuals I_{ir} and I_{jr} $(i \neq j)$ in a working individual set, denote their failure times as t_1 and t_2 $(t_1 < t_2)$ respectively. Let t'_1 and t'_2 represent the definitive replacement times for individuals I_{ir} and I_{jr}, respectively. Suppose $t'_1 = \eta_1$ and $t'_2 = \eta_2$, and there is an optimal structure W^* that has $\eta_1 > \eta_2$. The goal is to show that we can find a new optimal group structure which satisfies $t'_1 <= t'_2$ at the same or a lower cost.

Consider a group structure W' where both individuals are replaced at t'_1. It is obvious that individual I_{jr} is preventively replaced in W^*, and delay its replacement time to η_1 does not change its replacement type, meaning no additional cost because of no change in the replacement type. If individual I_{jr} is grouped with some other individuals in W^*, it is obvious that the cost of group structure W' is the same as W^*. If individual I_{jr} is not grouped with any other individual in W^*, then we eliminate one setup cost in group structure W', which leads to a lower cost.

Proof completed.

(A.3) The time complexity of the heuristic algorithm (Algorithm 4) is polynomial.

Proof:

The inputs that related to the time complexity are (1) not-used-residual lifetime of an individual $|\boldsymbol{\Delta}|$, (2) shifting window $|\boldsymbol{\iota}|$, (3) end of planning horizon T and (4) number of component n.

Denote $G(\cdot)$ as the running time function.

The running time regarding different n can be evaluated by the number of group structures found in Grouping Rule. For the 1st group structure, it requires $(n-1)$ steps. For the 2nd group structure, it requires $(n-2)$ steps. So forth, for the $(n-1)$ group structures that found with a given n, the total steps are $(n-1)+(n-2)+\ldots+1 = n(n-1)/2$. Therefore, the running time in terms of n is

$$G(n) = n(n-1)/2 + (n-1)c_1$$

where c_1 is a constant that represents the execution time of other statement in Grouping Rule.

In the worst case, the Grouping Rule is executed at every time point, i.e. T times. Thus, the running time in terms of n and T is

$$G(n,T) = TG(n) + Tc_2$$

where c_2 is a constant that represents the execution time of other statement.

Because Δ and ι are the search variables, the total running time is

$$G(n,T,|\boldsymbol{\Delta}|,|\boldsymbol{\iota}|) = |\boldsymbol{\Delta}|\cdot|\boldsymbol{\iota}|\cdot G(n,T) = n^2T|\boldsymbol{\iota}| + nT|\boldsymbol{\iota}|c_3 + T|\boldsymbol{\iota}|c_4$$

where $c_3 = 2c_1 - 1$ and $c_4 = 2(c_2 - c_1)$. Notice that $|\boldsymbol{\Delta}| = 2$ based on Theorem 1. Using big-O notation, the time complexity of this algorithm is $O(n^2T|\boldsymbol{\iota}|)$, which is polynomial.

Proof completed.

Reference

Alkhamis, T. M. and J. Yellen (1995). Refinery units maintenance scheduling using integer programming. *Applied Mathematical Modelling* 19(9): 543-549.

Amaran, S., T. Zhang, N. V. Sahinidis, B. Sharda and S. J. Bury (2016). Medium-term maintenance turnaround planning under uncertainty for integrated chemical sites. *Computers & Chemical Engineering* 84: 422-433.

Archibald, Y. and R. Dekker (1996). Modified block-replacement for multiple-component systems. *IEEE transactions on reliability* 45(1): 75-83.

Assaf, D. and J. G. Shanthikumar (1987). Optimal group maintenance policies with continuous and periodic inspections. *Management Science* 33(11): 1440-1452.

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

Bérenguer, C., A. Grall and B. Castanier (2000). Simulation and evaluation of condition-based maintenance policies for multi-component continuous-state deteriorating systems. *Proceedings of the foresight and precaution conference*.

Besnard, F., M. Patrikssont, A.-B. Strombergt, A. Wojciechowskit and L. Bertling (2009). An optimization framework for opportunistic maintenance of offshore wind power system. *PowerTech, 2009 IEEE Bucharest*, IEEE.

Birge, J. R. and F. Louveaux (2011). Introduction to stochastic programming, (Springer Science & Business Media).

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

Castanier, B., A. Grall and C. Bérenguer (2005). A condition-based maintenance policy with non-periodic inspections for a two-unit series system. *Reliability Engineering & System Safety* 87(1): 109-120.

Cowing, M. M., M. E. Paté-Cornell and P. W. Glynn (2004). Dynamic modeling of the tradeoff between productivity and safety in critical engineering systems. *Reliability Engineering & System Safety* 86(3): 269-284.

Cui, L. and H. Li (2006). Opportunistic maintenance for multi-component shock models. *Mathematical Methods of Operations Research* 63(3): 493-511.

Dekker, R. and P. A. Scarf (1998). On the impact of optimisation models in maintenance decision making: the state of the art. *Reliability Engineering & System Safety* 60(2): 111-119.

Dekker, R., R. E. Wildeman and F. A. Van der Duyn Schouten (1997). A review of multi-component maintenance models with economic dependence. *Mathematical Methods of Operations Research* 45(3): 411-435.

Dekker, R., R. E. Wildeman and R. Van Egmond (1996). Joint replacement in an operational planning phase. *European Journal of Operational Research* 91(1): 74-88.

Ding, F. and Z. Tian (2012). Opportunistic maintenance for wind farms considering multi-level imperfect maintenance thresholds. *Renewable Energy* 45: 175-182.

Epstein, S. and Y. Wilamowsky (1985). Opportunistic replacement in a deterministic environment. *Computers & operations research* 12(3): 311-322.

Gade, D., G. Hackebeil, S. M. Ryan, J.-P. Watson, R. J.-B. Wets and D. L. Woodruff (2016). Obtaining lower bounds from the progressive hedging algorithm for stochastic mixed-integer programs. *Mathematical Programming* 157(1): 47-67.

Goyal, S. and M. Kusy (1985). Determining economic maintenance frequency for a family of machines. *Journal of the Operational Research Society* 36(12): 1125-1128.

Goyal, S. K. and A. Gunasekaran (1992). Determining economic maintenance frequency of a transport fleet. *International Journal of Systems Science* 23(4): 655-659.

Hariga, M. (1994). A deterministic maintenance-scheduling problem for a group of non-identical machines. *International Journal of Operations & Production Management* 14(7): 27-36.

Hart, W. E., C. D. Laird, J.-P. Watson, D. L. Woodruff, G. A. Hackebeil, B. L. Nicholson and J. D. Siirola (2012). Pyomo-optimization modeling in python, (Springer).

Hart, W. E., J.-P. Watson and D. L. Woodruff (2011). Pyomo: modeling and solving mathematical programs in Python. *Mathematical Programming Computation* 3(3): 219.

Huynh, K. T., A. Barros and C. Bérenguer (2015). Multi-level decision-making for the predictive maintenance of k -out-of-n : F deteriorating systems. *IEEE transactions on Reliability* 64(1): 94-117.

Koochaki, J., J. A. Bokhorst, H. Wortmann and W. Klingenberg (2012). Condition based maintenance in the context of opportunistic maintenance. *International Journal of Production Research* 50(23): 6918-6929.

Laggoune, R., A. Chateauneuf and D. Aissani (2009). Opportunistic policy for optimal preventive maintenance of a multi-component system in continuous operating units. *Computers & Chemical Engineering* 33(9): 1499-1510.

Laporte, G. and F. V. Louveaux (1993). The integer L-shaped method for stochastic integer programs with complete recourse. *Operations research letters* 13(3): 133-142.

Nguyen, K.-A., P. Do and A. Grall (2015). Multi-level predictive maintenance for multi-component systems. *Reliability Engineering & System Safety* 144: 83-94.

Nicolai, R. P. and R. Dekker (2008). Optimal maintenance of multi-component systems: a review. Complex system maintenance handbook, (Springer)**:** 263-286.

Okumoto, K. and E. Elsayed (1983). An optimum group maintenance policy. *Naval Research Logistics (NRL)* 30(4): 667-674.

Paté‐Cornell, M. E. (1993). Learning from the piper alpha accident: A postmortem analysis of technical and organizational factors. *Risk Analysis* 13(2): 215-232.

Patriksson, M., A.-B. Strömberg and A. Wojciechowski (2015). The stochastic opportunistic replacement problem, part II: a two-stage solution approach. *Annals of Operations Research* 224(1): 51-75.

Pham, H. and H. Wang (2000). Optimal (τ , T) opportunistic maintenance of a k‐out‐of‐n: G system with imperfect PM and partial failure. *Naval Research Logistics (NRL)* 47(3): 223-239.

Ponchet, A., M. Fouladirad and A. Grall (2010). Assessment of a maintenance model for a multi-deteriorating mode system. *Reliability Engineering & System Safety* 95(11): 1244-1254.

Rao, A. N. and B. Bhadury (2000). Opportunistic maintenance of multi‐equipment system: a case study. *Quality and Reliability Engineering International* 16(6): 487-500.

Rardin, R. L. and R. L. Rardin (2016). Optimization in operations research, (Prentice Hall).

Rockafellar, R. T. and R. J.-B. Wets (1991). Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of operations research* 16(1): 119-147.

Scarf, P. A. (1997). On the application of mathematical models in maintenance. *European Journal of operational research* 99(3): 493-506.
Shafiee, M. and M. Finkelstein (2015). An optimal age-based group maintenance policy for multi-unit degrading systems. *Reliability Engineering & System Safety* 134: 230-238.
Sule, D. R. and B. Harmon (1979). Determination of coordinated maintenance scheduling frequencies for a group of machines. *AIIE Transactions* 11(1): 48-53.
Tan, J. S. and M. A. Kramer (1997). A general framework for preventive maintenance optimization in chemical process operations. *Computers & Chemical Engineering* 21(12): 1451-1469.
Thomas, L. (1986). A survey of maintenance and replacement models for maintainability and reliability of multi-item systems. *Reliability Engineering* 16(4): 297-309.
Tian, Z. and H. Liao (2011). Condition based maintenance optimization for multi-component systems using proportional hazards model. *Reliability Engineering & System Safety* 96(5): 581-589.
van Dijkhuizen, G. and A. van Harten (1996). Two-stage maintenance: a generalized age maintenance policy. *T&M working paper*(UT-TBK. OMST. WP. 96.05).
Van Horenbeek, A. and L. Pintelon (2013). A dynamic predictive maintenance policy for complex multi-component systems. *Reliability Engineering & System Safety* 120: 39-50.
Vos de Wael, S. (1995). A decision support system for LampRemplace.
Vu, H. C., P. Do, A. Barros and C. Bérenguer (2014). Maintenance grouping strategy for multi-component systems with dynamic contexts. *Reliability Engineering & System Safety* 132: 233-249.
Watson, J.-P. and D. L. Woodruff (2011). Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. *Computational Management Science* 8(4): 355-370.
Wildeman, R. E. and R. Dekker (1997). Dynamic influences in multi‐component maintenance. *Quality and reliability engineering international* 13(4): 199-207.
Wildeman, R. E., R. Dekker and A. Smit (1997). A dynamic policy for grouping maintenance activities. *European Journal of Operational Research* 99(3): 530-551.
Zhu, Q., H. Peng and G.-J. van Houtum (2015). A condition-based maintenance policy for multi-component systems with a high maintenance setup cost. *Or Spectrum* 37(4): 1007-1035.