NPS-LM-13-C10P05R08-066

EXCERPT FROM THE
PROCEEDINGS
OF THE
TENTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM
LOGISTICS MANAGEMENT

Improving Multi-Component Maintenance Acquisition With a Greedy Heuristic Local Algorithm

Sifat Kalam and Kash Barker, University of Oklahoma
Jose Emmanuel Ramirez-Marquez, Stevens Institute of Technology

Published April 1, 2013

The research presented in this report was supported by the Acquisition Research Program of the Graduate School of Business & Public Policy at the Naval Postgraduate School.

Preface & Acknowledgements

Welcome to our Tenth Annual Acquisition Research Symposium! We regret that this year it will be a “paper only” event. The double whammy of sequestration and a continuing resolution, with the attendant restrictions on travel and conferences, created too much uncertainty to properly stage the event. We will miss the dialogue with our acquisition colleagues and the opportunity for all our researchers to present their work. However, we intend to simulate the symposium as best we can, and these *Proceedings* present an opportunity for the papers to be published just as if they had been delivered. In any case, we will have a rich store of papers to draw from for next year’s event scheduled for May 14–15, 2014!

Despite these temporary setbacks, our Acquisition Research Program (ARP) here at the Naval Postgraduate School (NPS) continues at a normal pace. Since the ARP’s founding in 2003, over 1,200 original research reports have been added to the acquisition body of knowledge. We continue to add to that library, located online at www.acquisitionresearch.net, at a rate of roughly 140 reports per year. This activity has engaged researchers at over 70 universities and other institutions, greatly enhancing the diversity of thought brought to bear on the business activities of the DoD.

We generate this level of activity in three ways. First, we solicit research topics from academia and other institutions through an annual Broad Agency Announcement, sponsored by the USD(AT&L). Second, we issue an annual internal call for proposals to seek NPS faculty research supporting the interests of our program sponsors. Finally, we serve as a “broker” to market specific research topics identified by our sponsors to NPS graduate students. This three-pronged approach provides for a rich and broad diversity of scholarly rigor mixed with a good blend of practitioner experience in the field of acquisition. We are grateful to those of you who have contributed to our research program in the past and encourage your future participation.

Unfortunately, what will be missing this year is the active participation and networking that has been the hallmark of previous symposia. By purposely limiting attendance to 350 people, we encourage just that. This forum remains unique in its effort to bring scholars and practitioners together around acquisition research that is both relevant in application and rigorous in method. It provides the opportunity to interact with many top DoD acquisition officials and acquisition researchers. We encourage dialogue both in the formal panel sessions and in the many opportunities we make available at meals, breaks, and the day-ending socials. Many of our researchers use these occasions to establish new teaming arrangements for future research work. Despite the fact that we will not be gathered together to reap the above-listed benefits, the ARP will endeavor to stimulate this dialogue through various means throughout the year as we interact with our researchers and DoD officials.

Affordability remains a major focus in the DoD acquisition world and will no doubt get even more attention as the sequestration outcomes unfold. It is a central tenet of the DoD’s Better Buying Power initiatives, which continue to evolve as the DoD finds which of them work and which do not. This suggests that research with a focus on affordability will be of great interest to the DoD leadership in the year to come. Whether you’re a practitioner or scholar, we invite you to participate in that research.

We gratefully acknowledge the ongoing support and leadership of our sponsors, whose foresight and vision have assured the continuing success of the ARP:

- Office of the Under Secretary of Defense (Acquisition, Technology, & Logistics)
- Director, Acquisition Career Management, ASN (RD&A)
- Program Executive Officer, SHIPS
- Commander, Naval Sea Systems Command
- Program Executive Officer, Integrated Warfare Systems
- Army Contracting Command, U.S. Army Materiel Command
- Office of the Assistant Secretary of the Air Force (Acquisition)
- Office of the Assistant Secretary of the Army (Acquisition, Logistics, & Technology)
- Deputy Director, Acquisition Career Management, U.S. Army
- Office of Procurement and Assistance Management Headquarters, Department of Energy
- Director, Defense Security Cooperation Agency
- Deputy Assistant Secretary of the Navy, Research, Development, Test, & Evaluation
- Program Executive Officer, Tactical Aircraft
- Director, Office of Small Business Programs, Department of the Navy
- Director, Office of Acquisition Resources and Analysis (ARA)
- Deputy Assistant Secretary of the Navy, Acquisition & Procurement
- Director of Open Architecture, DASN (RDT&E)
- Program Executive Officer, Littoral Combat Ships

James B. Greene Jr.
Rear Admiral, U.S. Navy (Ret.)

Keith F. Snider, PhD
Associate Professor

Logistics Management

Political, Legal, and Economic Considerations

John Fallon, *University of Maryland, University College*
Timothy Reed, *Beyond Optimal Strategic Solutions*

Optimizing Causes of Procurement Cost Through Strategic Sourcing: The Impact of Rate, Process, and Demand

Timothy Reed, *Beyond Optimal Strategic Solutions*
Michael E. Knipper, *United States Air Force*
John Fallon, *University of Maryland, University College*

Improving Multi-Component Maintenance Acquisition With a Greedy Heuristic Local Algorithm

Sifat Kalam—Kalam received her BS in industrial engineering from the University of Oklahoma in 2010 and will complete her MS degree in May 2013. She is currently on educational leave of absence from The Boeing Company and will be joining Boeing Defense, Space & Security (BDS) as a supply chain specialist after completion of her MS degree. Her current research is in heuristics and meta-heuristics based algorithms to model acquisition and preventive maintenance frameworks for multi-component systems. [lipa0335@ou.edu]

Kash Barker—Barker is an assistant professor in the School of Industrial and Systems Engineering at the University of Oklahoma. Dr. Barker and his students in the Risk-Based Decision Making Laboratory are primarily interested in (i) modeling the reliability, resilience, and interdependent economic impacts of disruptions to critical infrastructure networks and (ii) enhancing data-driven decision making for large-scale system sustainment. He received his PhD in systems engineering from the University of Virginia, where he worked in the Center for Risk Management of Engineering Systems. [kashbarker@ou.edu]

Jose Emmanuel Ramirez-Marquez—Ramirez-Marquez is an associate professor in the School of Systems and Enterprises at the Stevens Institute of Technology and the director of the School's Engineering Management program. As director of the Systems Development and Maturity Laboratory, Dr. Ramirez-Marquez's work advances systems management and assessment for optimal development of a system through its lifecycle. His other interests include reliability analysis, network resilience, and optimization. He received his PhD in industrial engineering from Rutgers University.

Abstract

As many large-scale DoD systems age, and due to budgetary and performance efficiency concerns, there is a need to improve the decision making process for system sustainment, including maintenance, repair, and overhaul (MRO) operations and the acquisition of MRO parts. To help address the link between sustainment policies and acquisition, this work develops a greedy heuristic–based local search algorithm to provide a system maintenance schedule for multi-component systems, coordinating recommended component maintenance times to reduce system downtime costs thereby enabling effective acquisition.

Introduction

Large organizations such as the Department of Defense (DoD) have to devote a significant amount of their budgets to system maintenance. According to a 2007 Government Accountability Office (GAO) report, the DoD spends approximately 40% of its budget on operations and management (O&M) activities to ensure system readiness ($209.5 billion in 2005). GAO reported that since fiscal year 2001, the DoD's O&M costs are increasing, and the Air Force, in particular, had to increase its O&M cost by 29%. As many large-scale DoD systems age, and due to budgetary and performance efficiency concerns, there is a need to improve the decision making process for system sustainment, including maintenance, repair, and overhaul (MRO) operations and the acquisition of MRO parts.

The DoD's acquisition costs have seen growth in recent years (GAO, 2013). The GAO (2013) recommended that the DoD improve its strategic management plan to make maintenance supply chain operations more cost effective. Further, the DoD was advised to "link acquisition and sustainment policies" for depot maintenance improvement and ultimate cost efficiency (GAO, 2011). To help address the link between sustainment policies and acquisition, this work develops a framework to provide a system maintenance schedule for multi-component systems. As the multiple components of a system have their own lifecycles, an efficient means to schedule overall system maintenance should consider these individual components to maximize long-term availability of the system. This framework coordinates recommended maintenance times, such as those found as a result of reliability centered maintenance (RCM) or from original equipment manufacturer (OEM) suggestions, to formulate a system-level maintenance schedule for a finite time horizon. Such a framework will increase the acquisition efficiency of components with a more effective system-level maintenance schedule.

With the recent computational advances, several preventive maintenance models have been proposed for complex multi-component systems considering component interactions. In the preventive maintenance scheduling problem (PMSP), different kinds of component interactions are taken into account. Interaction among components can be economic dependence, structural dependence, and/or stochastic dependence (Thomas, 1986). In a basic sense, economic dependence among system components means that the cost of joint repair is different from cost of individual repair (Dekker, Wildeman, & van der Duyn Schouten, 1997), suggesting that performing repair operations for multiple components at once can be done with less expense than for single components.

Researchers have considered different model formulations, as well as solution techniques, to address preventive maintenance decision making. Stinson and Khumawala (1987) formulated a heuristics-based mixed integer linear program (MILP) model for a finite horizon preventive maintenance problem for maintaining machines in series. Budai, Huisman, and Dekker (2006) proposed a heuristics-based MILP solution for scheduling railroad network maintenance. Other few noteworthy approaches are Bayesian network model (Celeux, Corset, Lannoy, & Ricard, 2006), goal programming for a multi-objective problem (Bertolini & Bevilacqua, 2006), and dynamic programming (Dekker, Wildeman, & Van Egmond, 1996).

In terms of algorithm development, Dekker, Smit, and Losekoot (1991) presented an optimal maintenance model using a set-partitioning algorithm for multiple maintenance activities. One downside of their model was that they considered each activity time to be negligible relative to the total planning horizon. Later Dekker et al. (1996) solved the above-mentioned problem with a dynamic programming formulation, concluding that the dynamic algorithm is a good heuristic for rolling horizon–based problems which can incorporate short-term system information for decision support. Dekker et al. (1997) provided a review of maintenance models for multi-component systems, which covered economically dependent systems. The Markov decision chain–based approach was also studied by Dekker et al. (1996) for the multi-activities maintenance problem which was applicable to systems consisting of many components. Previous Markov chain–based models were limited to few components. An opportunistic maintenance policy was modeled by Gürler and Kaya (2002) and van der Duyn Schouten and Vanneste (1993) for identical multi-component systems. Sheu et al. (1996) modeled a similar kind of problem with a two-stage opportunistic policy. In the case of non-identical components maintenance, the tradeoff between the repair cost of one component versus another should be considered, including the resulting increase in the complexity of the model.

PMSP remains a very active area of research. Little work in this field has used heuristics and meta-heuristics based methodologies to model preventive maintenance framework (Nicolai & Dekker, 2008). A new meta-heuristic based on a genetic algorithm was applied in train maintenance scheduling problems by Sriskandarajah, Jardine, and Chan (1998), primarily optimizing cost. Nicolai and Dekker (2008) presented a review of preventive maintenance and recommended that more researches need to be done in this area developing more heuristic and meta-heuristic approaches like simulated annealing and local search. Meta-heuristic based algorithms have proven very successful for flowshop scheduling problems (Pan & Ruiz, 2012), which have similar characteristics to preventive maintenance scheduling.

This work presents a greedy heuristic–based local search algorithm for preventive maintenance of multiple components which would be a new contribution in this field of research. We develop a local search–based algorithm to minimize the total maintenance cost of a system over a finite planning horizon. This paper is organized as follows. The Methodological Development section provides a detailed description of the different components and procedure of our proposed schedule algorithm for a multi-component system. The next section, Greedy Heuristic with Local Search Algorithm, provides experimental results for a presented multi-component scheduling problem. We conclude our paper with the Experimental Results section and some concluding remarks.

Methodological Development

Here we develop a new formulation and solution algorithm to address preventive maintenance scheduling for a multi-component system. It is assumed that maintenance results in a "good as new" condition.

Baseline individual component maintenance times for planning horizon T (i.e., system-in-use time) are known and recommended based on a mean time between failure (MTBF) calculation (e.g., by RCM or OEM calculations). We assume these component maintenance times are given in their in-use-time or up-time. Our goal is to suggest to alter the recommended maintenance schedule for a multi-component system in a joint manner for as many components as possible. Performing many individual maintenance events at recommended schedules can potentially lead to cost savings due to reduced setup costs and reduced downtime. However, varying too far from recommended MTBF guidance can lead to unnecessary maintenance (in the earliness situation) and risk of failure (in the tardiness situation). *Earliness* refers to the performance of maintenance earlier than recommended, with *tardiness* representing the performance of maintenance at a time later than recommended. As such, there are penalties associated with both earliness and lateness, as well as a penalty for system downtime while maintenance is being performed.

Different potential maintenance schedules can be compared and evaluated using a penalty function approach (Yousefi & Yosuff, 2013). In this approach, a penalty function can be achieved by quantifying setup-related costs into setup penalties, downtime costs into downtime penalties, related expense (i.e., costs of unnecessary maintenance) of earliness into earliness penalties, and potential failure costs of tardiness situation into tardiness penalties. By implementing this approach, a maintenance schedule can be found which will minimize these penalties. These penalties, as well as other notation, are defined as follows:

T	Planning horizon
n	Number of components in the system
C_S	System setup penalty per maintenance
$T_{m,l}^{k}$	kth maintenance time for component l
$C_{E,l}$	Earliness penalty for component l, per unit time
$C_{L,l}$	Tardiness penalty for component l, per unit time
C_D	System downtime penalty per unit time
$T_{r,l}$	Component maintenance duration for component l
δ	Construction phase time-span parameter where $\delta \in (0, 1]$
γ	Joint maintenance time parameter $\gamma \in (0, 1]$
Δ_j	Deviation of individual component maintenance times from jth system maintenance
T_{max}	Maximum time-span of construction phase
T_c	Construction phase time-span
T_{m1}	Set of first component maintenance time
T_{m2}	Set of second component maintenance time
π^c	Candidate solution
π^d	Discard solution
S^c	Candidate combination set
S^d	Discard set
S	Algorithm solution vector

Decision variables for the scheduling formulation include the following:

t_m^j	jth system maintenance time
R	Total number of system maintenance events scheduled in planning horizon T
x_l^j	If feature earliness is present in component l for maintenance j ($x_l^j = 1$) or not ($x_l^j = 0$)
y_l^j	If feature tardiness is present in component l for maintenance j ($y_l^j = 1$) or not ($y_l^j = 0$)
z_l^j	If component l should be repaired at time t_m^j ($z_l^j = 1$) or not ($z_l^j = 0$)

Performing joint repair has the potential to save maintenance cost because for many multi-component systems it is possible to perform component maintenance simultaneously. Thus total repair time for joint maintenance depends on individual instance and can be predicted from previous system maintenance data. Considering all these penalties, our goal is to develop an algorithm that will schedule system maintenance time such that total penalties of system maintenance are minimized over the given planning horizon *T*. The basic optimization problem is conceptualized in Equation 1, where $C_S R$ represents total setup penalties for planning horizon *T*, and C_l^j represents penalties associated with *j*th system repair of component *l*. C_l^j includes penalties for downtime, earliness, and tardiness for component *l* during *j*th system maintenance. Decision variable z_l^j determines whether component l will be repaired at *j*th system maintenance.

$$\min_{z_l^j} C_S R + \sum_{l=1}^{n} \sum_{j=1}^{R} C_l^j z_l^j$$
$$\text{s.t.} \quad z_l^j \in \{0,1\} \tag{1}$$
$$\text{MRO requirement constraints}$$

Equation 2 presents the actual objective function and constraints for the problem above.

$$\min_{t_m^j} C_S R + \sum_{l=1}^{n} \sum_{j}^{R} \left|T_{m,l}^k - t_m^j\right| C_{E,l} x_l^j + \sum_{l=1}^{n} \sum_{j}^{R} \left(T_{m,l}^k - t_m^j\right)^2 C_{L,l} x_l^j + \sum_{l}^{n} \sum_{j}^{R} C_D \gamma T_{r,l} z_l^j$$
$$\begin{aligned} \text{s.t.} \quad & R > 0 \\ & t_m^j > 0 \\ & x_l^j \in \{0,1\} \\ & y_l^j \in \{0,1\} \\ & z_l^j \in \{0,1\} \\ & \gamma \in (0,1] \end{aligned} \tag{2}$$

One of the decision variables is the system-in-use time at which system maintenance should be performed. As maintenance scheduling is multistage (e.g., maintenance is a repeated event), the time at which maintenance is scheduled for iteration *j* is t_m^j This work will solve Equation 2 over a finite time horizon for several MRO stages, finding a series of t_m^j values at which maintenance should occur. OEM-recommended individual component maintenance times are denoted by $T_{m,l}^k$. Here t_m^j values attempt to coordinate the downtime of several components to maximize long-term availability of the system. Equation 2 conceptualizes an availability cost problem, where z_l^j determines whether component *l* should be repaired at time t_m^j according to the cost function which penalizes unavailability. Equation 2 also attempts to improve upon $T_{m,l}^k$ to minimize the deviation of individual component maintenance times from system maintenance time, found in the neighborhood of $T_{m,l}^k$. As such, this work provides the maintenance schedule for the system, whether the *j*th maintenance operation will repair an optimal subset of the *n* components in the system.

Elements of the above formulation are given more detail as follows. The actual structure of the penalty function here can vary due to decision maker preferences.

Penalty Function

Our objective is to minimize total system maintenance penalty over a finite time horizon *T*. Our penalty function is the presented objective function in Equation 2. This total penalty function consists of system setup penalty, system downtime time penalty, and penalty for any deviation of individual component maintenance times from system maintenance time. Note that we are not penalizing for the cost of performing actual repair, including the cost of acquisition and the cost of labor, among others, under the assumption that this cost is the same for individual repair and joint repair.

System Setup Penalty

The setup penalty component in Equation 3 accounts for the time to arrange for system maintenance. A system setup penalty penalizes all associated costs for maintenance setup, charged only once regardless of the number of multiple components involved in a maintenance work. Not included is component setup time, as that is not expected to be a factor in determining individual or joint maintenance; any maintenance performed on a component would require component setup time. Fixed system penalty per maintenance work C_S is known.

$$\text{System setup penalty} = C_S\, R \tag{3}$$

Earliness Penalty

There is a penalty for executing the component maintenance at a time other than the maintenance recommended by the OEM. If system maintenance is scheduled earlier than recommended individual component maintenance, then there is a penalty for early maintenance work for that component. This penalty attempts to penalize the performance of maintenance unnecessarily too far in advance of the OEM recommendation, and it is a function of (i) the total amount of earliness determined by $\left|T_{m,l}^{k} - t_m^j\right|$ (ii) the earliness penalty $C_{E,l}$, and (iii) whether component *l* maintenance is performed early, determined by x_l^j.

$$\text{Earliness penalty} = \sum_{l=1}^{n} \sum_{j}^{R} \left|T_{m,l}^{k} - t_m^j\right| C_{E,l}\, x_l^j \tag{4}$$

Tardiness Penalty Cost

If system maintenance is scheduled later than individual component maintenance, then there is a penalty for late maintenance work for that component. This penalty is a function of the deviation of recommended individual component maintenance times from the actual system maintenance time. The penalty is higher for tardiness than earliness here due to aversion to performing maintenance later than recommended. This is represented, in part, by the square on the amount of tardiness time, $\left(T_{m,l}^{k} - t_m^j\right)^2$. Other elements include tardiness penalty $C_{L,l}$ and whether component *l* maintenance is performed after the OEM suggested maintenance time, determined by y_l^j.

$$\text{Tardiness penalty} = \sum_{l=1}^{n} \sum_{j}^{R} \left(T_{m,l}^{k} - t_m^j\right)^2 C_{L,l}\, y_l^j \tag{5}$$

System Downtime Cost

There is a cost associated with system downtime due to an unproductive or idle system. The system downtime penalty per unit time C_D is known. Expected component maintenance duration for component *l* is parameterized as $T_{r,l}$. Parameter γ represents the percentage of total expected component maintenance duration (i.e., $\sum T_{r,l}$ for all *l* that are present in *j*th system maintenance) that would be the expected joint maintenance duration for *j*th system repair. We assume this γ value to be constant for all iterations. The value of joint maintenance time parameter γ can be chosen from the historical data of a related system such that $\gamma \in (0, 1]$. The higher the γ parameter value, the higher the downtime maintenance cost would be. Higher γ means less time savings in joint repair compared to separate maintenance. γ reaches a value of 1 when the expected joint repair time is equal to the summation of individual component repair times; those are present in *j*th joint repair. In other words, the expected downtimes are the same for joint repair and separate repair when $\gamma = 1$.

This value defines joint maintenance times for a multi-component system. The term z_l^j determines whether the *j*th maintenance operation for component *l* is performed.

$$\text{System downtime cost} = \sum_{l}^{n} \sum_{j}^{R} C_D \gamma \, T_{r,l} z_l^j \tag{6}$$

Construction Phase Time-Span Parameter (δ)

At the beginning, construction phase time-span parameter delta ($\boldsymbol{\delta}$) is chosen such that $\boldsymbol{\delta} \in (0, 1]$. This $\boldsymbol{\delta}$ value is kept constant throughout the algorithm. Discussed later, the algorithm solution is very sensitive to this delta value and needs to be tuned according to individual instance. A detailed sensitivity analysis and tuning recommendation of $\boldsymbol{\delta}$ are presented later.

Weibull Distribution

The recommended individual maintenance times are assumed here to be the MTBF from a two-parameter Weibull distribution. The Weibull distribution is well known in reliability analysis in describing the time between failures for components. MTBF for a Weibull distribution is found in Equation 7, where β is the shape parameter, η is the scale parameter, and Γ is the gamma function.

$$\text{MTBF} = \eta \Gamma(1 + \frac{1}{\beta}) \tag{7}$$

Greedy Heuristic With Local Search Algorithm

The maintenance optimization model described previously is solved with a proposed iterative Greedy Heuristic with Local Search Algorithm (GHLSA). The proposed algorithm is similar to the generic structure of the Greedy Randomized Adaptive Search Procedure (GRASP; Feo & Resende, 1995]. In contrast to the two phases of GRASP, our proposed algorithm has three phases: (1) a construction phase, (2) an improvement phase, and (3) a local search phase. In the GRASP algorithm, the initial solution is constructed using a randomized sampling technique, whereas our algorithm uses a greedy heuristic to construct an initial partial solution. We also use an additional improvement phase, where the greedy heuristic–based improvement ends. An overview of the proposed algorithm is presented in Figure 1.

```
procedure GHLSA ()
    begin
        I ← InputInstance { };
        for GHLSA stopping criterion not satisfied →
            π_j^0 ← InitialPartialSolution (I,δ);
            π_j' ← GHBI (π_j^0);
            π_j'' ← LocalSearch (π_j');
            UpdateSolution (π_j'');
        endfor
        return OptimalSolutionFound;
    end GHLSA;
```

Figure 1. Pseudo-Code Overview of the Proposed Greedy Heuristic With Local Search Algorithm (GHLSA)

In brief, the three phases of the algorithm achieve the following:

1. *The construction* phase determines how many components in the system should be initially examined to include in system maintenance of multiple components and an initial estimate for the time at which that multi-component maintenance operation should occur.
2. *The improvement phase* improves the construction phase result by dividing the set of multiple components into two sets (a *candidate set* and a *discard set*) to determine whether dividing the maintenance operation will produce a lower penalty than the construction phase. This phase iterates by removing a component out of the candidate set one at a time and placing it in the discard set and calculating penalty improvement.
3. *The local search phase* focuses on the resulting candidate set from the improvement phase and iterates across the different times associated with recommended component maintenance to balance the penalties of earliness and tardiness of individual components.

These three phases are performed at each iteration *j*, thereby resulting in the set of components involved in the *j*th system maintenance operation and the time at which the *j*th system maintenance operation should be performed. The algorithm stopping criterion is the pre-determined planning horizon *T*. Let *I* be the set of discrete time periods where each element represents recommended (e.g., from RCM or OEM suggestions) repair times of a component during planning horizon *T*.

The final solution of this algorithm is essentially an *R* × *1* vector for all system maintenance operations, where each element of the vector represents the recommended *j*th system maintenance. The result of each iteration *j* is referred to as the *j*th partial solution of the over final solution. Each element of the algorithm solution is comprised of two parts: π_j [0] refers to the set of repair times $\left\{T_{m,\mathrm{A}_1}^{a_1}, \dots, T_{m,\mathrm{A}_n}^{a_n}\right\}$ of components to be performed jointly at the *j*th system maintenance operation (where $T_{m,\mathrm{A}_n}^{a_n}$ is the a_n maintenance operation for component A_n), and π_j[1] refers to the recommended time t_m^j at which the *j*th system maintenance is to be performed. For example, $\pi_j = \left[\pi_j[0], \pi_j[1]\right] = \left[\{T^a_{m,\mathrm{A}}, T^b_{m,\mathrm{B}}, T^c_{m,\mathrm{C}}\}, t^j_m\right]$ suggests that the *a*th maintenance operation of component A, the *b*th maintenance of component B, and the *c*th maintenance of component C will all be performed jointly at time t^j_m, the time chosen for the *j*th system maintenance operation to occur. Thus during each iteration of this algorithm, it finds an element which we refer to as a partial solution for algorithm solution set. At each iteration *j*, the three phases of the algorithm are performed, each of which is explained in detail subsequently. Through these three phases of construction and improvement, a partial solution is found, and this partial solution is then added to the solution set to update the algorithm solution for the scheduling maintenance problem. This iterative process is completed when the solution is found for the given planning horizon.

Using input instance *I* and chosen value δ, an initial partial solution π^0_j is created in the construction phase. During the improvement phase, this initial partial solution π^0_j is improved using greedy heuristic–based procedure GHBI. This improved partial solution is represented by π'_j.. During the local search phase of the jth iteration, partial solution π'_j.is further improved using the LocalSearch procedure, and the third phase returns the final partial solution π''_j. After finding the best partial solution π''_j in the third phase, we need to update the existing algorithm solution *S* and input set *I*. This partial solution '' is then added as the jth element to solution vector *S*, to update the algorithm solution. All scheduled component maintenance times $T^k_{m,l}$ at iteration *j* are removed from set *I* for the next $(j + 1)$st iteration, and the rest of the unscheduled component repair times of set *I* are updated according to their earliness or tardiness deviation for *j*th system maintenance.

Phase 1: Initial Partial Solution Construction

At each iteration *j*, the first phase is a construction phase where the initial partial solution is generated. General pseudo-code for this partial solution construction phase is presented in Figure 2. $T_{\max}$ is the time duration which expresses the maximum time-span which includes all the component repair times to be initially considered for repair during jth system maintenance. The construction phase time-span is selected according to the δ value, which reduces the length of time originally considered by proportion δ. All component repair times $T^k_{m,l}$during time-span T_c are included in the joint repair component set for the initial partial solution π^0_j for iteration *j*. This constructs the first part of the initial partial solution, π^0_j[0].

Step 1.1. Calculate T_{max}

The maximum time-span of construction phase $T_{\max}$ needs to be calculated. This $T_{\max}$ value represents the time duration between the recommended time for the earliest first repair of all components and the recommended time for the earliest second repair. Let the sets of first and second repair times of each component out of all unscheduled maintenance times be T_{m1} and T_{m2}, respectively. The minimum value of set T_{m1} is denoted by EarliestFirstRepairTime, and the minimum value of set T_{m2} is expressed by EarliestSecondRepairTime in the pseudo-code in Figure 2. The absolute value of their difference is the value of time-span $T_{\max}$

Step 1.2. Calculate T_c

Construction phase time-span $T_c$can be calculated by multiplying the value of the maximum time-span of construction phase $T_{\max}$ by δ. In a sense, δ is the scope of granularity. A small value of δ suggests a tight granularity of the maintenance option set, meaning that a shorter time frame will be considered for T_c with which to consider multiple component maintenance options. For a larger value of δ, $T_{\max}$ approaches $T_{\max}$ value. And T_c is equal to $T_{\max}$ when $\delta = 1$.

Step 1.3. Partial Solution Component Set

Insert all recommended component maintenance times $T_{m,l}^{k}$ that are originally scheduled during construction phase time-span T_c into the joint repair component set $\pi_j^0[0]$ of the initial partial solution π_j^0. If there are n_{m1} elements in set T_{m1}, then it would take n_{m1} iterations to construct the initial partial solution component set.

The time at which system maintenance is performed on the components in $\pi_j^0[0]$ constitutes the second part of the initial partial solution, $\pi_j^0[1]$, which can be chosen according to several heuristics including

- the mid-point of time-span T_c,
- a component repair time of component set $\pi_j^0[0]$ where the deviation Δ_j is minimized, or
- the earliest component repair time (i.e., the minimum value of component set π_j^0 [0]).

In our implementation, the third heuristic above is used to construct the later part of the initial partial solution. That is, the second part of the initial partial solution, π_j^0 [1], is chosen according to the heuristic convention of scheduling system repairs at the earliest component repair time. Thus, this phase schedules all possible component maintenance during time-span T_c at the earliest possible time to produce an initial partial solution.

```
procedure InitialPartialSolution (I,δ)
    begin
        π_j^0 ← { };
        T_max ← |EarliestFirstRepairTime - EarliestSecondRepairTime |;
        n_m1 ← |T_m1|
        T_c ← δ * T_max
        for i ← 1 to n_m1 do
            if T_m1[i] < T_c then
                π_j^0 ← π_j^0 U T_m1[i] ;
            endif
        endfor
        return π_j^0;
    end InitialPartialSolution;
```

Figure 2. Pseudo-Code for GHLSA Phase 1, the Partial Solution Construction Phase

Phase 2: Greedy Heuristic-Based Improvement (GHBI)

During the second phase of iteration *j*, the algorithm improves the initial partial solution π_j^0 constructed in Phase 1, focusing primarily on the components in $\pi_j^0[0]$ to be repaired jointly (e.g., $\{T_{m,\mathrm{A}}^{a}, T_{m,\mathrm{B}}^{b}, T_{m,\mathrm{C}}^{c}\}$). A search is performed in the neighborhood of π_j^0 to find a better partial solution. This combination of component repair times is improved according to a greedy heuristic of removing the last-one-out (i.e., latest component repair time) from existing combinations.

Let the initial partial solution π_j^0 be the existing partial solution π_j' (i.e., *j*th solution element). If there are n_p elements in the joint repair component set (π_j^0[0]) of the existing partial solution, then there would be n_p possible combinations of component sets that can be created according to the last-one-out greedy heuristic. The best combination set among n_p possible combinations is selected in $(n_p - 1)$ iterations. At each iteration of the $(n_p - 1)$, two temporary partial solution elements called *candidate solution* π^c and *discard solution* π^d (i.e., temporary *j*th and (*j*+1)st) are generated from existing partial solution π_j'. The best candidate solution is selected as the new existing partial solution π_j' according to an acceptance criterion. Each iteration of this greedy heuristic–based improvement method, which is the ImproveCombination procedure in Figure 3, is described below.

Step 2.1. Determining $\boldsymbol{\pi_j'[0]}$

The first part of a solution element presents the component repair times to be repaired jointly. Improved combination of this joint repair component set is searched using the last-one-out heuristic. To generate an improved combination of the *j*th solution element, two sets (i.e., candidate combination set S_c and discard set S_d) are created from the existing joint repair component set. The candidate set will eventually be repaired during the *j*th iteration, and the discard set will be saved for the (*j* + 1)st iteration or beyond. Let the existing joint repair component set be the initial value of candidate combination set S_c. By applying the last-one-out greedy heuristic (i.e., latest component repair time), a new discard set S_d is created. To generate the discard set S_d, the latest component repair time (i.e., max S_c) is removed from candidate solution set S_c and inserted into discard set S_d. Candidate set S_c and discard set S_d construct the first part of the candidate solution π^c and discard solution π^d respectively (i.e., $\pi^c[0]$ and $\pi^d[0]$).

Step 2.2. Determining $\boldsymbol{\pi_j'[1]}$

The time at which the elements of the candidate solution $\pi^c[0]$ are repaired is found from the earliest component repair time heuristic for the set (i.e., min S_c). This time of repair is $\pi^c[1]$. Similarly, the components in discard solution $\pi^d[0]$ are repaired at $\pi^d[1]$, or min S_d. Other heuristics that could be used in this step were presented in step 3 of the previous phase.

Step 2.3. Acceptance Criterion

The candidate solution is selected as the existing partial solution π_j', according to the acceptance criterion of the minimum penalty function. The existing candidate solution is chosen as the partial solution π_j' if the combined penalty function value of candidate and discard solutions is less than the penalty function value of the existing partial solution π_j'.

Figure 3 presents the procedure of developing new combination set according to the greedy heuristic.

```
procedure ImproveCombination ( π_j^0 )
   begin
         CurrentPenalty ← PenaltyFunction (π_j^0);
         π_j'  ← π_j^0 ;
         π_c  ← [  ] ;
         π_d  ← [  ] ;
         S_c ← π_j^0[ 0 ];
         S_d  ← {  };
         n_p  ← |π_j^0[ 0 ]|;
         for i ← 1 to (n_p -1) do
               S_c  ←  remove last component repair time and insert it in  S_d ;
               π_c  ← [{ S_c }, min ( S_c ) ] ;
               π_d  ← [{ S_d }, min ( S_d ) ] ;
               NewPenalty ←  PenaltyFunction (π_c )+ PenaltyFunction (π_d);
               if NewPenalty < CurrentPenalty then                 % Acceptance criterion
                     π_j'   ← π_c  ;
                     CurrentPenalty ← PenaltyFunction (π_c  ) ;
               endif
         endfor
         return  π_j' ;
   end  ImproveCombination ;
```

Figure 3. Pseudo-Code for Improving Combination Stage

As long as the number of elements n_p of existing partial solution π_j' is greater than 1 and minimizes the penalty function value, π_j' is divided into two new parts: candidate solution π^c and discard solution π^d. This iterative improvement is performed in the *while* loop presented in procedure GHBI. Figure 4 describes the procedure GHBI using pseudo-code.

```
procedure GHBI (π_j^0)
    begin
      π_j' ← π_j^0 ;
      CurrentPenalty ← PenaltyFunction (π_j^0);
      n_p ← |π_j^0[0]|;
      NewPenalty ← 0;
      while (NewPenalty < CurrentPenalty and n_p > 1 ) do
          CurrentPenalty ← PenaltyFunction ( π_j' );
          π_j' ← ImproveCombination ( π_j' );          % Using greedy heuristic last-one-
          out
          NewPenalty ← PenaltyFunction ( π_j' );
            n_p ← | π_j'[0]| ;
      endwhile
      return π_j' ;
    end GHBI;
```

Figure 4. Pseudo-Code for GHLSA Phase 2, the Greedy Heuristic-Based Improvement Phase

Phase 3: Local Search-Based Improvement

In the last phase of system maintenance iteration *j*, an improved partial solution is selected by searching the neighborhood of current partial solution π_j', building the best candidate set of components repair at the *j*th iteration. Let this improved partial solution be π_j'' and its initial value be π_j'. Emphasis in this third phase is placed primarily on searching different values of t_m^j in the neighborhood of $\pi_j'[1]$ to determine when the *j*th maintenance operation should occur. The pseudo-code for this local search phase is shown in Figure 5. During this improvement phase, t_m^j iteratively takes the values of component maintenance time generated from the final combination set $\pi_j'[0]$ during the previous phase and creates a temporary partial solution. During this iterative process, the partial solution is updated according to the penalty function in Equation 2. According to our selected method, it takes n_p iterations to search the neighborhood of $\pi_j'[1]$, if the number of elements in combination set $\pi_j'[0]$ is n_p. At each iteration of n_p, a new temporary partial solution called *temp* is generated. Steps of each iteration are as follows.

Step 3.1. Determining $\pi_j''[0]$

The joint repair component set comprising $\pi_j''[0]$ takes the value of the final combination set (i.e., $\pi_j'[0]$) found in the second phase.

Step 3.2. Determining $\pi_j''[1]$

During this improvement phase $temp[1]$ (i.e., t_m^j) iteratively takes the values of the component maintenance time generated from the final combination set $\pi_j'[0]$ during the previous phase. At iteration n_p, t_m^j would take the value of n_pth element of combination set $\pi_j'[0]$.

Step 3.3. Acceptance Criterion

The acceptance criterion is the value of the penalty function presented in Equation 2. At each iteration of n_p, the temporary partial solution *temp* is selected as the new existing partial solution only if the new temporary partial solution minimizes the penalty function value.

At the end of n_p iterations, the LocalSearch procedure returns the best value found in the search. The return value, π_j'', of this local search–based improvement is the partial solution representing the *j*th element of the final solution vector.

```
procedure LocalSearch (π'_j)
    begin
        π''_j ← π'_j ;
        CurrentPenalty ← PenaltyFunction (π'_j);
        NoOfElement ← |π'_j [0]|;
        if NoOfElement  > 1 then
            for i ← 1 to NoOfElement do
                temp ← π''_j ;
                temp [1] ← π'_j [0] [ i ];
                NewPenalty ← PenaltyFunction (temp);
                if NewPenalty < CurrentPenalty  then
                    π''_j [1] ← π'_j [0] [ i ];
                    CurrentPenalty = PenaltyFunction (π''_j);
                endif;
            endfor;
        endif;
        return π''_j ;
    end LocalSearch;
```

Figure 5. Pseudo-Code for the GHLSA Phase 3, the Local Search Phase

Experimental Results

An example problem briefly illustrates the algorithm.

Problem Specification

Our example problem addresses 10 components in a multi-component system. We assume the initial start time TNOW is zero. We assumed the earliness penalty and tardiness penalty values to be equal and same for all components (i.e., deviation penalty C_p). Maintenance duration $T_{r,l}$ is assumed to be 5 time units for all components. The recommended individual maintenance times of these components are assumed here to be the MTBF from a two-parameter Weibull distribution with shape parameters (β) and scale parameters (η). The assumed values of planning horizon, setup cost, downtime cost per unit time, earliness penalty, and tardiness penalty are presented in Table 1.

Original Case

The original case follows a simple procedure for maintenance. Each system maintenance operation is performed at the earliest component repair time (i.e., min $T^k_{m,l}$) out of unscheduled component maintenance times. It is assumed that all repair times are in the system repair window (i.e., min [$T^k_{m,l}$ + $T_{r,l}$]) and will be scheduled to be repaired at the same time. We used the same penalty function to calculate the objective function value for the original case. Note that the tardiness penalty will always be zero in the original case instance, as system maintenance is done at the earliest component repair time and there is no push back of component maintenance.

Table 1. Parameters of the Illustrative Example

Component	η	β	Other values
A	15	2	TNOW =0
B	20	3	T = 200 time unit
C	15	3	C_S =30,000
D	17	4	C_D =5,000
E	23	5	C_p=$C_{E,l}$= $C_{L,l}$=500, for all l
F	37	4	$T_{r,l}$= 5500, for all l
G	30	7	
H	22	3	
I	19	2	
J	26	4	

Experimental Evaluation

We solved the above-mentioned problem with our proposed algorithm (GHLSA) and performed a comparative study between the original case results and GHLSA results. Generated experimental penalty function data were transformed into percent deviation value (PD). We calculated the PD of objective function value resulting from proposed algorithm implementation, from the original case result using the following equation, where $\text{Obj}_{\text{Original}}$ represents penalty function value for original case and $\text{Obj}_{\text{GHLSA}}$ represents penalty function value produced using GHLSA procedure. Positive PD means the objective function value has improved (i.e., minimized) using the proposed algorithm and vice versa.

$$\text{Percentage Deviation (PD)} = \frac{\text{Obj}_{\text{Original}} - \text{Obj}_{\text{GHLSA}}}{\text{Obj}_{\text{Original}}} \times 100 \tag{8}$$

All calculated results for given instance for different delta values are presented in Table 2.

Sensitivity Analysis on γ

Table 2 shows that for a given instance, the proposed algorithm produced a very high objective function value which resulted in negative PD value for lower γ value (i.e., γ = 0.1 to γ = 0.3). For γ value greater than 0.3, calculated PD resulted in positive values. So for higher γ values (i.e., for γ > 0.3), the best solutions found using proposed GHLSA improved the objective function value of the original case. As γ increased, the PD value decreased for both positive and negative deviation trends. This trend was true for any γ value (Figure 6). Collected data were not very sensitive to γ value. Trend of the PD remained the same, and objective function value changed a little bit with a change in γ.

Sensitivity Analysis on δ

For all γ values, the objective function percent deviation change was logarithmic with δ (Figure 6). For lower values of δ, GHLSA produced some negative deviation. As δ increased, it generated positive deviation, as the objective function value decreased with higher δ value.

Table 2. Objective Function and PD Values for Given Instance

δ		γ=0.1	γ=0.2	γ=0.3	γ=0.4	γ=0.5	γ=0.6	γ=0.7	γ=0.8	γ=0.9	γ=1
	Original	1100101.52	1355101.52	1610101.52	1865101.52	2120101.52	2375101.52	2630101.52	2885101.52	3140101.52	3395101.52
0.1	GHLSA	-799063	-799063	-799063	-799063	-799063	-799063	-799063	-799063	-799063	-799063
	PD	-72.64	-58.97	-49.63	-42.84	-37.69	-33.64	-30.38	-27.70	-25.45	-23.54
0.2	GHLSA	-165297	-165297	-165297	-165297	-165297	-165297	-165297	-165297	-165297	-165297
	PD	-15.03	-12.20	-10.27	-8.86	-7.80	-6.96	-6.28	-5.73	-5.26	-4.87
0.3	GHLSA	-74705	-74705	-74705	-74705	-74705	-74705	-74705	-74705	-74705	-74705
	PD	-6.79	-5.51	-4.64	-4.01	-3.52	-3.15	-2.84	-2.59	-2.38	-2.20
0.4	GHLSA	61451	61451	61451	61451	61451	61451	61451	61451	61451	61451
	PD	5.59	4.53	3.82	3.29	2.90	2.59	2.34	2.13	1.96	1.81
0.5	GHLSA	119326	119326	119326	119326	119326	119326	119326	119326	119326	119326
	PD	10.85	8.81	7.41	6.40	5.63	5.02	4.54	4.14	3.80	3.51
0.6	GHLSA	186958	186958	186958	186958	186958	186958	186958	186958	186958	186958
	PD	16.99	13.80	11.61	10.02	8.82	7.87	7.11	6.48	5.95	5.51
0.7	GHLSA	172660	172660	172660	172660	172660	172660	172660	172660	172660	172660
	PD	15.69	12.74	10.72	9.26	8.14	7.27	6.56	5.98	5.50	5.09
0.8	GHLSA	226176	226176	226176	226176	226176	226176	226176	226176	226176	226176
	PD	20.56	16.69	14.05	12.13	10.67	9.52	8.60	7.84	7.20	6.66
0.9	GHLSA	226176	226176	226176	226176	226176	226176	226176	226176	226176	226176
	PD	20.56	16.69	14.05	12.13	10.67	9.52	8.60	7.84	7.20	6.66
1	GHLSA	226176	226176	226176	226176	226176	226176	226176	226176	226176	226176
	PD	20.56	16.69	14.05	12.13	10.67	9.52	8.60	7.84	7.20	6.66

Comparative Study

We performed a comparative study of the original case and the GHLSA-based results by generating different instances by changing given value of C_s, C_D, and C_P.

Sensitivity Analysis on Setup Penalty C_s

Produced results for the original case and the GHLSA case for different generated instances for six different setup costs are presented in Table 3. The presented values are the calculated PD values for the best objective function value found using the proposed GHLSA for each instance.

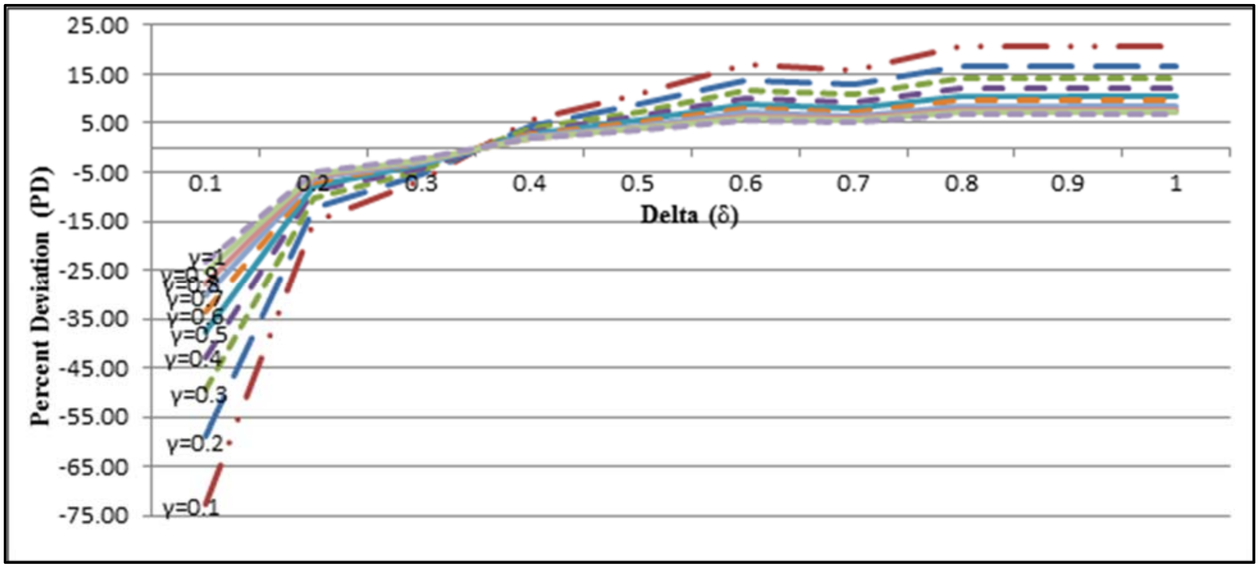

Figure 6. Change in PD Value With Delta

For all generated 60 instances, GHLSAs were able to improve (i.e., positive PD values) the original case penalty function value (Table 3). Improvement ranged from 1.08% to 20.56% in minimizing the objective function value compared to the original case. For a given C_s, penalty function value increased as γ decreased. It showed an increasing trend in PD value with increasing C_s, for any given γ. It shows the potential of this research algorithm for multi-component system maintenance where setup cost is comparatively high.

Table 3. PD Values for Different Setup Costs

Setup	γ=0.1	γ=0.2	γ=0.3	γ=0.4	γ=0.5	γ=0.6	γ=0.7	γ=0.8	γ=0.9	γ=1
30k	20.56	16.69	14.05	12.13	10.67	9.52	8.60	7.84	7.20	6.66
25k	18.07	14.32	11.86	10.12	8.83	7.83	7.03	6.38	5.84	5.39
20k	14.84	11.42	9.28	7.81	6.75	5.94	5.30	4.79	4.37	4.01
15k	10.51	7.77	6.17	5.11	4.37	3.81	3.38	3.03	2.75	2.52
10k	6.39	4.49	3.46	2.81	2.37	2.05	1.80	1.61	1.45	1.33
5k	6.32	4.12	3.05	2.42	2.01	1.72	1.50	1.33	1.19	1.08

Sensitivity on Downtime Penalty C_D

We generated 100 instances for 10 different C_D values ranging from 1k to 10k. The calculated PD values are representative of the best solution found using proposed GHLSA at granularity level 0.1 (Table 4). The proposed GHLSAs were able to improve the PD values of all 100 instances for different C_D. PD values ranged from 3.80 to 25.24. For all γ, PD value decreased with higher C_D.

Table 4. PD Values for Different Downtime Costs

Downtime Cost	γ=0.1	γ=0.2	γ=0.3	γ=0.4	γ=0.5	γ=0.6	γ=0.7	γ=0.8	γ=0.9	γ=1
1k	25.24	23.88	22.66	21.56	20.56	19.65	18.82	18.05	17.34	16.69
2k	23.88	21.56	19.65	18.05	16.69	15.52	14.51	13.62	12.83	12.13
3k	22.66	19.65	17.34	15.52	14.05	12.83	11.80	10.93	10.18	9.52
4k	21.56	18.05	15.52	13.62	12.13	10.93	9.95	9.13	8.44	7.84
5k	20.56	16.69	14.05	12.13	10.67	9.52	8.60	7.84	7.20	6.66
6k	19.65	15.52	12.83	10.93	9.52	8.44	7.57	6.87	6.28	5.79
7k	18.82	14.51	11.80	9.95	8.60	7.57	6.76	6.11	5.57	5.12
8k	18.05	13.62	10.93	9.13	7.84	6.87	6.11	5.50	5.01	4.59
9k	17.34	12.83	10.18	8.44	7.20	6.28	5.57	5.01	4.55	4.16
10k	16.69	12.13	9.52	7.84	6.66	5.79	5.12	4.59	4.16	3.80

Sensitivity on Deviation Penalty C_P

Different C_P values, ranging from 100 to 1,000, were used to generate 100 experimental instances. All calculated PD values of the original case and the GHLSA case are in Table 5.

Table 5. PD Values for Different Deviation Penalty Values

Deviation Penalty	γ=0.1	γ=0.2	γ=0.3	γ=0.4	γ=0.5	γ=0.6	γ=0.7	γ=0.8	γ=0.9	γ=1
100	27.85	22.30	18.59	15.94	13.95	12.41	11.17	10.15	9.31	8.59
200	25.93	20.84	17.42	14.96	13.11	11.67	10.51	9.56	8.77	8.10
300	24.08	19.41	16.27	14.00	12.28	10.94	9.86	8.98	8.24	7.62
400	22.29	18.03	15.14	13.05	11.47	10.23	9.23	8.41	7.72	7.14
500	20.56	16.69	14.05	12.13	10.67	9.52	8.60	7.84	7.20	6.66
600	18.89	15.39	12.98	11.22	9.88	8.83	7.98	7.28	6.69	6.19
700	17.28	14.12	11.93	10.33	9.11	8.15	7.37	6.73	6.19	5.73
800	15.72	12.88	10.91	9.46	8.35	7.48	6.77	6.18	5.69	5.27
900	14.42	11.85	10.06	8.74	7.72	6.92	6.27	5.73	5.27	4.89
1000	12.75	10.51	8.93	7.77	6.88	6.17	5.59	5.11	4.71	4.37

In all experimental instances for deviation penalty C_P, our presented GHLSAs were successfully able to minimize the penalty function value. For all 100 instances, the found PD values were positive, which means improvement of the objective function value compared to the original case study. For different C_P values, the resulting PD values ranged from 4.37 to 27.85. PD value decreased with higher C_P for all γ (see Table 5).

Optimal δ Value

Tables 6 and 7 present the optimal δ values at granularity level 0.1, for generated instances for C_S and C_P. Note that optimal δ values for downtime instances were not reported in the tables. For all 100 instances for C_D, the generated optimal delta value was 0.8–1.0 at granularity level 0.1. For setup cost, all instances for cost ranging from 15k to 30k, optimal δ was 0.8–1.0. For 10k setup cost instances, the optimal delta value was 1.0, and it decreased to 0.5 for the lowest setup cost 5k.

Table 6. Optimal δ Values for Different Setup Cost Instances

SetupCost	γ=0.1	γ=0.2	γ=0.3	γ=0.4	γ=0.5	γ=0.6	γ=0.7	γ=0.8	γ=0.9	γ=1
5k	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
10k	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
15k–30k	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0

Table 7. Optimal δ Values for Different Deviation Penalty Cost Instances

DeviationPenalty	γ=0.1	γ=0.2	γ=0.3	γ=0.4	γ=0.5	γ=0.6	γ=0.7	γ=0.8	γ=0.9	γ=1
100–800	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0
900	0.7–1.0	0.7–1.0	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7
1000	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0	0.8–1.0

For deviation penalty instances, this optimal δ value was constant for all values of C_P, except 900. For C_P = 900, for lower γ values (i.e., γ = 0.1–0.2), the δ value for best found results was 0.7–1.0 and for the rest of γ values, it was 0.7. According to the PD analysis, δ is a significant factor in finding a good solution by implementing this greedy heuristic–based methodology. The experimental results of the presented 260 instances shows that a higher value of δ, at granularity level 0.1, is a safer choice when the penalty function for all the δ values cannot be evaluated. In those cases, our recommended δ value would be 0.5–1.0, at granularity level 0.1.

Tuning Parameter δ

The presented results were very sensitive to δ. Tuning of this granularity parameter depends on the system configuration (i.e., the number of components) and available computation power. If possible, the initial tuning can be done at granularity level 0.1. Granularity level 0.1 means to change the scope of granularity δ value by 0.1. With a granularity level of 0.1, in 10 runs the algorithm would generate the best solution possible with the proposed method. If the granularity level needs to be smoother, that depends on the input instance (i.e., the input values of $T_{m,l}^{k}$). If $T_{m,l}^{k}$ values result in a very small T_{max}, then a higher granularity level may not produce any better result, as a number of components repair times may remain the same for resulting construction phase time-span T_c.

Concluding Remarks and Future Work

In this paper, we proposed a greedy heuristic local search algorithm for multi-component preventive maintenance scheduling problems. This scheduling algorithm is based on some greedy heuristics and a local search method. This new algorithm has proven to make significant improvement of the objective function criterion, compared to presented original case results. We have implemented the presented GHLSA for 260 generated instances and found remarkable results. Deviation analysis showed significant improvement of the objective function value for all 260 problem instances. The presented greedy heuristics–based algorithm looks very promising in solving some real life preventive maintenance scheduling problems.

Future work includes the addition of another objective to the algorithm: the effect on system reliability at iteration *j*. Currently, only a cost parameter is considered when determining the earliness or tardiness of a particular component maintenance operation when coordinating system maintenance. However, it is hypothesized that a system reliability objective may change the maintenance schedule, particularly when the system schedule suggests that some components be maintained after their recommended maintenance times (tardiness), potentially resulting in an undesired system reliability.

References

Bertolini, M., & Bevilacqua, M. (2006). A combined goal programming–AHP approach to maintenance selection problem. *Reliability Engineering and System Safety, 91*(7), 839–848.

Budai, G., Huisman, D., & Dekker, R. (2006). Scheduling preventive railway maintenance activities. *Journal of the Operational Research Society, 57*(9), 1035–1044.

Celeux, G., Corset, F., Lannoy, A., & Ricard, B. (2006). Designing a Bayesian network for preventive maintenance from expert opinions in a rapid and reliable way. *Reliability Engineering and System Safety*, *91*(7), 849–856.

Dekker, R., Smit, A., & Losekoot, J. (1991). Combining maintenance activities in an operational planning phase: A set-partitioning approach. *IMA Journal of Management Mathematics, 3*(4), 315–331.

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

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

El-Amin, I., Duffuaa, S., & Abbas, M. (2000). A Tabu Search algorithm for maintenance scheduling of generating units. *Electric Power Systems Research, 54*(2), 91–99.

Feo, T. A., & Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, *6*(2), 109–133.

GAO. (2007). *Defense budget: Trends in operation and maintenance costs and support services contracting* (GAO-07-631). Washington, DC: Author.

GAO. (2011). *Defense logistics: DOD input needed on implementing depot maintenance study recommendations* (GAO-13-267). Washington, DC: Author.

GAO. (2013). *Defense business transformation: Improvements made but additional steps needed to strengthen strategic planning and assess progress* (GAO-13-267). Washington, DC: Author.

Gürler, Ü., & Kaya, A. (2002). A maintenance policy for a system with multi-state components: An approximate solution. *Reliability Engineering and System Safety, 76*(2), 117–127.

Higgins, A. (1998). Scheduling of railway track maintenance activities and crews. *Journal of the Operational Research Society, 49*(10), 1026–1033.

Nicolai, R. P., & Dekker, R. (2008). Optimal maintenance of multi-component systems: A review. In K. A. H. Kobbacy & D. N. P. Murthy (Eds.), *Complex system maintenance handbook* (pp. 263–286). London, England: Springer London.

Pan, Q. K., & Ruiz, R. (2012). Local search methods for the flowshop scheduling problem with flowtime minimization. *European Journal of Operational Research, 222*(1), 31–43.

Papadakis, I. S., & Kleindorfer, P. R. (2005). Optimizing infrastructure network maintenance when benefits are interdependent. *OR Spectrum, 27*(1), 63–84.

Sriskandarajah, C., Jardine, A., & Chan, C. (1998). Maintenance scheduling of rolling stock using a genetic algorithm. *Journal of the Operational Research Society, 49*(11), 1130–1145.

Stinson, J. P., & Khumawala, B. M. (1987). The replacement of machines in a serially dependent multi-machine production system. *International Journal of Production Research, 25*(5), 677–688.

van der Duyn Schouten, F. A., & Vanneste, S. G. (1993). Two simple control policies for a multicomponent maintenance system. *Operations Research, 41*(6), 1125–1136.

Visser, J. K. (1998). Modelling maintenance performance: A practical approach. In *Proceedings of IMA Conference* (pp. 1–13). Edinburgh, Scotland.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL
555 DYER ROAD, INGERSOLL HALL
MONTEREY, CA 93943

www.acquisitionresearch.net