Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Vehicle Fleet Maintenance Scheduling by Multi-objective Evolutionary Algorithms

Yali Wang, Steffen Limmer, Markus Olhofer, Michael Emmerich, Thomas Bäck

2019

Preprint:

This is an accepted article published in 2019 IEEE Congress on Evolutionary Computation (CEC). The final authenticated version is available online at: https://doi.org/10.1109/CEC.2019.8790142

Vehicle Fleet Maintenance Scheduling Optimization by Multi-objective Evolutionary Algorithms

1st Yali Wang
LIACS, Leiden University
Leiden, The Netherlands
y.wang@liacs.leidenuniv.nl

2nd Steffen Limmer
Honda Research Institute Europe GmbH
Offenbach, Germany
steffen.limmer@honda-ri.de

3rd Markus Olhofer
Honda Research Institute Europe GmbH
Offenbach, Germany
markus.olhofer@honda-ri.de

4th Michael T.M. Emmerich
LIACS, Leiden University
Leiden, The Netherlands
m.t.m.emmerich@liacs.leidenuniv.nl

5th Thomas Bäck
LIACS, Leiden University
Leiden, The Netherlands
t.h.w.baeck@liacs.leidenuniv.nl

***Abstract*—In this paper, a new real-world application problem, i.e., the vehicle fleet maintenance scheduling optimization problem, is defined and a specialized multi-objective evolutionary algorithm framework (grouping strategy, three vector chromosome and corresponding genetic operators) is proposed to solve the problem. State-of-the-art multi-objective evolutionary algorithms such as NSGA-III, SMS-EMOA, DI-MOEA are employed in the proposed algorithm framework to solve the problem, and their behavior is investigated. Although DI-MOEA is used the first time for a real-world application problem, its performance is better than other algorithms for some instances.**

***Index Terms*—multi-objective optimization, evolutionary algorithm, maintenance optimization, scheduling optimization**

I. Introduction

Nowadays, companies, corporations, and organizations of all sorts rely on vehicle fleets to deliver products and services. Typical examples of such vehicle fleets are taxi cab fleets, public bus fleets, car rental fleets, delivery fleets, and so on. According to the statistical data from the European Automobile Manufacturers Association, the global vehicle fleet grows continuously, and the EU has a total fleet of 259.7 million passenger cars and 39.1 million commercial vehicles in 2017 [1]. The maintenance of fleet vehicles plays a critical role in their efficient use. Fleet vehicles should be maintained according to a schedule to ensure that they are safe for use; at the same time, neglecting fleet maintenance may lead to higher costs and shorter lives for vehicles.

Many businesses choose to outsource vehicle maintenance because it is not their core business functions. For a fleet maintenance service provider, ideally, a complete process of making the maintenance schedule for a vehicle fleet is: first, data are collected from the sensors on the fleet vehicles [2], then sensor data are analyzed to acquire the condition of the car/component. E.g., the remaining useful lifetime (RUL) of the component is predicted based on adequate predictive approaches or models [3]. After that, the predicted RUL or other knowledge about the condition of the car/component is used to optimize the maintenance schedule for the vehicle fleet. At the same time, the business requirements are taken into account in the scheduling optimization. In this way, the obtained maintenance schedule(s) can ensure that all maintenance tasks are performed on time, keeping the vehicle fleet in healthy operating condition and under business requirements.

This paper formulates a real-world scheduling optimization problem and focuses on the optimization of the maintenance schedule of a vehicle fleet based on the predicted RULs of components. Section 2 introduces the problem formulation. A literature review is provided in Section 3. In Section 4, we propose a specialized multi-objective evolutionary algorithm framework to solve the problem. Section 5 presents and discusses experimental results. Section 6 concludes the work and outlines some relevant future work.

II. Problem Formulation

The size of a vehicle fleet can be large and its distribution wide, which makes it necessary to distribute the maintenance of a vehicle fleet in multiple separate workshops. While each workshop has only limited teams and each team can repair only one car at the same time. Besides, each workshop is limited to the maintenance of specific components due to the restriction in equipment or the skill level of the staff. These form the primary constraints faced by maintenance scheduling optimization. Furthermore, it is possible that the maintenance of the same component produces different costs and workloads when the operation is performed in different workshops because, for example, the distances between vehicles and the workshops are different. Fixed set-up cost and set-up time are considered for each visit of a car to a workshop, which correspond to the cost and time required for the preparation of the maintenance. In our present study, the predicted RUL is converted to the due date which is the estimate of the date when the component fails in case no maintenance takes place before its due date.

The vehicle fleet maintenance scheduling optimization problem addressed in this paper is a new application problem and defined as follows:

1. There are n cars $C = \{C_1, C_2, \cdots, C_n\}$ and m workshops $W = \{W_1, W_2, \cdots, W_m\}$.

2. Each car C_i comprises l_i components to be maintained for $i = 1, \cdots, n$.

3. For each component O_{ij}, i.e. the jth operation of car C_i, there is a set of workshops capable of repairing it. The set of workshops is represented by W_{ij} which is a subset of W.

4. The processing time of the maintenance of the component O_{ij} in workshop W_k is predefined and denoted by p_{ijk}.

5. The maintenance cost of the maintenance of the component O_{ij} in workshop W_k is predefined and denoted by q_{ijk}.

6. The set-up time of car C_i in workshop W_k is predefined and denoted by X_{ik}.

7. The set-up cost of car C_i in workshop W_k is predefined and denoted by Y_{ik}.

8. The number of teams in workshop W_k is predefined and denoted by Z_k.

9. The due date and previous repair time of each component O_{ij} are predefined and denoted by D_{ij} and R_{ij} respectively.

10. All business requirements or vehicle demands are predefined. There are r vehicle demands and the format of one demand is: N_i cars are required from day d_{i1} to day d_{i2} for $i = 1, \cdots, r$.

At the same time, the following assumptions are made:

1. All workshops and cars are available at time 0 and assumed to be continuously available.

2. All the components are independent from each other.

3. Times required for transport of cars from/to workshops are included in the maintenance time and cost of cars.

4. Environmental changes (such as car accidents) are not considered here.

5. There are no precedence constraints among the components of different cars. Cars are maintained on a first-come-first-served basis.

6. An operation, once started, must run to completion.

7. No operation can start before completion of the previous operation.

We assume that three objectives are relevant for the vehicle fleet operator, which are the total workload, total cost and demand satisfaction. The reason why demand satisfaction is defined as objective is that it is treated as flexible and violable. Let T_k denote the sum of the maintenance times spent on all operations that are processed in workshop W_k; M_i the sum of all costs of all operations of car C_i; N^t_{avail} the number of cars which are not in workshops on day t, N^t_{demand} the number of cars required on day t. Three objectives can be defined as:

$$\text{Minimum total workload} : f_1 = \sum_{k=1}^{m} T_k \tag{1}$$

$$\text{Minimum total cost} : f_2 = \sum_{i=1}^{n} M_i \tag{2}$$

Maximum demand satisfaction :

$$f_3 = \sum_{t} \min\{N^t_{avail} - N^t_{demand}, 0\} \tag{3}$$

For illustration purposes, an example of parameters for a car is shown in Table I, where rows correspond to components of the car; columns correspond to the cost, processing time, set-up time and set-up cost of the car in alternative workshops, also due date and previous repair date. In this example, there are two workshops: W_1 and W_2. The tag "$-$" means that the workshop cannot repair the corresponding component, therefore, workshop W_1 cannot maintain O_{14} and workshop W_2 cannot maintain O_{11}. Here, the predefined costs of components are the same at both workshops: €200, so are the processing times, set-up costs and set-up times. Of course, times and costs in different workshops can be different. The due date and previous repair time are relative to day 0.

TABLE I. Parameters of car C_1

Comp	cost		time		date		set-up cost		set-up time	
	W_1	W_2	W_1	W_2	Due	Pre	W_1	W_2	W_1	W_2
O_{11}	200	-	2	-	74	-61	100	1	100	1
O_{12}	200	200	2	2	25	-6				
O_{13}	200	200	2	2	15	-50				
O_{14}	-	200	-	2	60	-1				

III. Literature review

In the early studies of predictive maintenance, also called condition-based maintenance (CBM) [4], the condition monitoring, failure diagnostics, and prognostics always attracted more attention than planning the maintenance schedule based on the information obtained from the condition monitoring, failure diagnostics and prognostics. In [5], an onboard locomotive diagnostic system was invented to monitor the transmitted onboard vehicle data, determine whether any of the monitored data is out of a predetermined range, compare monitored data with historical data and calculate trends, predict if any vehicle system(s) must be corrected to avoid vehicle failure and when such system(s) are likely to fail. It was mentioned in [5] that the onboard diagnostic systems are not helpful in optimizing locomotive maintenance scheduling because they do not communicate with a rail carrier's scheduling center.

Simultaneously, the prognostic information included in maintenance policies may be given in different formats such as RUL, efficiency decrease, the probability of failure. Prognostics and reliability information in [6] were the failure probabilities of the components. The failure probabilities were analyzed to schedule maintenance with minimum system risk using a genetic algorithm. The systematic risk was treated as part of the cost of performing the maintenance schedule. System conflicts and resources were also considered as constraints in this single-objective problem. [7] introduced deterioration models to take into account the component degradation information of a multi-component system. The optimal dates of maintenance operations can be computed based on the model, and additional costs can be evaluated if the operations are not executed at the optimal maintenance dates. Lastly, the optimal grouping of individual maintenance actions was found to reduce the maintenance cost, and the maintenance plan was updated dynamically at each inspection date.

Optimizing the maintenance schedule has been widely neglected in the study of predictive maintenance, while it can

be crucial for effective maintenance planning and scheduling. The flexible job shop scheduling problem (FJSP) is an essential branch of production planning problems and can be suggestive for our maintenance scheduling optimization. The FJSP consists of a set of independent jobs to be processed on multiple machines, and each job contains several operations with a predetermined order. It is assumed that each operation must be processed in specified processing time on a specific machine out of multiple alternatives. The problem has been extensively studied in the literature, for example, [8], [9].

According to [10], most research papers have addressed classical FJSP in the last 25 years, while only around 35% of papers considered different scenarios such as machine breakdown, uncertain processing times, overlapping operations and so on. [11] considered FJSP-PPF (process plan flexibility), where jobs can have alternative process plans. It was assumed that the process plans are known in advance and that they are represented by linear precedence relationships. Because only one of the alternative plans had to be adopted for each job, the FJSP-PPF dealt with not only routing and sequencing sub-problems, but also the process plan selection sub-problem. In this paper, a mixed-integer linear programming model was developed for the FJSP-PPF and makespan was adopted as the single performance measure. While only hypothetical test problems with routing and process plan flexibility were used for the testing model since there is no comparable mathematical model in the literature.

Compared with classical FJSP, our problem has some special properties: (1) flexible sequence: the sequence of the components is not predefined, and the starting time of each component is mainly influenced by its due date. (2) multiple problem parameters: besides the processing time, other problem parameters like the maintenance cost, set-up time, set-up cost, repair teams, the demand for cars at a specific time, also have an impact on the result.

IV. Solution Methodology

Due to the combinatorial nature of the problem, heuristics and metaheuristics, in particular evolutionary algorithms (EAs), are the standard approaches to it. A specialized evolutionary algorithm framework is defined and applied with multi-objective evolutionary algorithms (MOEAs) to solve our multi-objective scheduling optimization problem. In this section, we detail the approach underlying the algorithm and the implementation of genetic search, including chromosome encoding, chromosome decoding and genetic operators.

A. Grouping Components

When scheduling components, the idea of grouping several components of the same car for one visit is employed in our algorithm. By grouping the maintenance of multiple components into one maintenance operation, the set-up cost and set-up time apply only once for the complete group of components. However, the maintenance cost could be indirectly penalized:

- with the reduction of the component useful life if the maintenance date is shifted backward;
- with the increase of the risk of breaking down on the road if the maintenance date is shifted forward.

Therefore, the maintenance of each component should not be shifted too far from its due date. In our algorithm, an interval or execution window of the starting time is defined for each component, and the maintenance of the component can only start at a time spot inside the corresponding interval. The interval of each component consists of two parts and their respective lengths are defined as:

- Length of the interval before the due date: $0.3\times$ (Due date - Previous repair date);
- Length of the interval after the due date: $0.1\times$ (Due date - Previous repair date).

The interval is chosen relatively long so that maintenance before or after the interval hardly makes sense. With the interval, combining components can only be effective if their intervals overlap, and the starting time of the group maintenance must lie within the interval intersection.

Grouping components also means that not all the components can be maintained exactly at their due dates. For the component that is maintained before or after its due date, an extra cost is introduced to penalize either the reduction of the useful life when the maintenance is performed before the due date or the increase of component failure probability when the maintenance is performed after the due date. Figure 1 gives an example of the execution window and penalty function of a component. Two linear penalty functions are proposed to calculate the penalty cost based on the following assumptions.

- If a component is maintained at the same time as the previous repair time, the penalty cost would be $c + s$: the sum of its maintenance cost and the set-up cost of the car;
- If a component is maintained at the end of the interval (the latest possible repair time), the penalty cost would be $100 * c$ (c: the maintenance cost of the component). This penalty cost is the combination of all losses: the expense needed if the defect occurs on the use (diagnostics, technical and logistics support, repair or replacement of the failed component), the loss of reputation, and so on.

As can be seen, if the component O_{ij} is maintained at t_{ij} (t_{ij} is in its interval), its penalty cost is:

$$\begin{cases} ((c+s)/(D_{ij}-R_{ij})) \times (D_{ij}-t_{ij}) & \text{if} \quad t_{ij} < D_{ij} \\ 0 & \text{if} \quad t_{ij} = D_{ij} \\ ((100 \times c)/((D_{ij}-R_{ij})/10)) \times (t_{ij}-D_{ij}) & \text{if} \quad t_{ij} > D_{ij} \end{cases}$$

When components are grouped together, the maintenance time of the group is the sum of the processing times of all components in the group plus one set-up time; the cost of the group is the combination of one set-up cost, the maintenance costs and the penalty costs of all components in the group.

B. Chromosome Encoding

EAs typically start with a diverse set of feasible solutions (a population) and iteratively replace the current population by a new population. A suitable encoding is required for

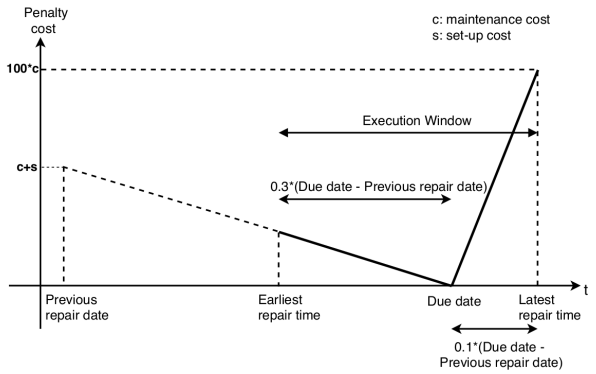

Fig. 1: Execution window and penalty function of a component

the problem. In our algorithm, a three vector chromosome (Figure 2) is proposed to represent an individual:

- Group structure vector: the group structures of vehicles one by one.
- Starting time vector: the starting times of group operations.
- Workshop assignment vector: the workshops of group operations.

1) Group structure vector: Figure 3 represents the intervals of eight components of a car. Component $c1$ can be grouped with $c2$ and/or $c3$ due to the overlap between their execution windows. Other possible group structures can be deduced in the same manner. However, the practical situation can be more complicated. For example, in Figure 4, $c1$ can be grouped with $c2$; $c1$ can be grouped with $c3$ and $c4$; $c1$ can also be grouped with $c3$ and $c5$. But $c1$ and $c2$ together cannot be grouped with other components; $c1$, $c3$ and $c4$ together cannot be grouped with $c5$. We check overlaps and possible group structures of all components from one car by the following sweep line algorithm:

1. All components of one car are sorted as a list according to the left end of the intervals.
2. Starting from the first component, all components are checked one by one. For each component, only this component and the subsequent components in the list are necessary in the check because the possible groups with the preceding components have been found when checking those components.
3. A stack is utilized to store the intervals and their corresponding components. The interval of the checked component is pushed onto the stack first. Look at the next component, if there is an overlap between them, push the overlap interval onto the stack; otherwise, pop off the topmost element from the stack, return the only group possibility and end the checking process.
4. Compare the topmost interval on the stack with the interval of the next component, if they overlap, push the overlap interval onto the stack; otherwise, pop off the topmost element from the stack. Repeat this step until the interval stack is empty or the last component has been checked.

All possible group numbers will be found after the above steps. Figure 5 shows the alternative groups of components on Figure 3 (left table) and Figure 4 (right table). Each component corresponds to a column and the numbers in its column are all the possible group numbers the component can choose. Components which have picked the same group number belong to the same group. Table II shows two randomly generated group structure vectors of the car in Figure 3. The first example "1 2 1 4 5 6 7 7" represents that $c1$ and $c3$ are in the same group; $c7$ and $c8$ are in the same group; $c2$, $c4$, $c5$, $c6$ are in four different groups. It does not matter what the group number is, the group number only tells us which components are in the same group. For example, "3 4 5 5 7" could be a group structure vector of the car in Figure 4.

TABLE II. Example of group structure

Component	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Example 1	1	2	1	4	5	6	7	7
Example 2	1	1	1	4	4	6	7	8

2) Starting time vector: A starting time vector can be randomly generated based on its group structure vector. Before picking the starting time for each group, the interval intersection of each group is calculated and the starting time is randomly picked from this intersection. Under the condition that a component is the sole member of a group, its starting time can be selected from its entire maintenance interval.

3) Workshop assignment vector: A workshop is considered as "several workshops" based on its capacity (the number of teams). By this way, the schedule of each workshop team can be achieved from the solution. For example, consider that two workshops have three and four repairing teams respectively. Then, group operations can be randomly assigned to seven "workshops", the former three and the latter four represent corresponding teams in two workshops.

C. Genetic Operators

Based on the encoding, the corresponding genetic operators are designed. Crossover is a matter of replacing some genes in one parent by the corresponding genes of the other. Crossover operators are applied separately to the three parts of the chromosomes.

For the group structure vector, one point or multi-point crossover can be used as crossover operator. Figure 6 is an example of one point crossover on the two group structure vectors in Table II. Two new vectors are generated, and they are always valid because the group number of each component is from its own alternative group numbers. Because the whole group structure vector consists of all car components, one point crossover may not be efficient enough. Therefore, multi-point crossover is applied.

The same cutting points are applied to the starting time vector when doing crossover. However, the change on the group structure vector caused by the crossover can result in the invalidity of the starting time vector because it is possible that the group members and intersection have changed due to the new group structure. Therefore, when performing the crossover on the starting time vector, the starting times of

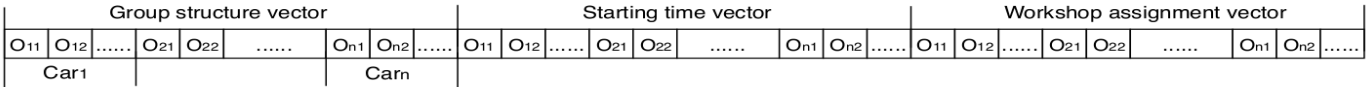

Fig. 2: Three vector chromosome

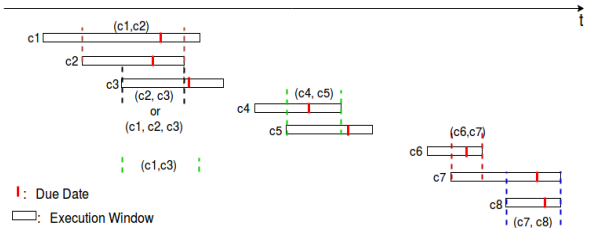

Fig. 3: Possible groups of a car with eight components

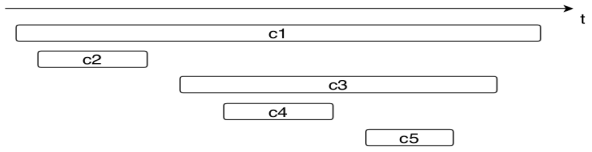

Fig. 4: Five component intervals of a car

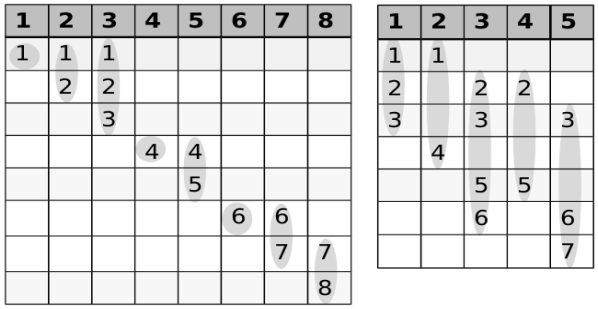

Fig. 5: Alternative groups of components in Fig.3 and Fig.4

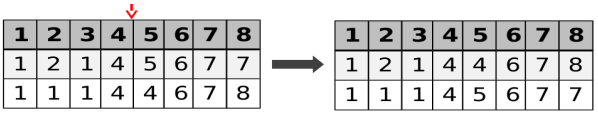

Fig. 6: One point crossover

all group operations are checked and a new starting time is generated randomly from the new feasible intersection in the case that the starting time of a group is invalid.

A multi-point crossover can be applied to the workshop assignment vector as well. Every component is assigned a workshop team randomly. In our algorithm, the workshop team of a group operation is decided by the first component in that group in case different workshop teams are assigned to components in the same group.

The mutation operator alters one or more gene values in a chromosome from its initial state. The mutation operators applied to three parts of the chromosome are also implemented separately. One or more gene values in the group structure vector can be exchanged by another alternative value from the same column in Figure 5 in order to generate a new individual. Again, the change applied to the group structure vector can result in the invalidity of the starting time vector. Hence, the starting time is checked for the corresponding groups after the mutation is done on the group structure vector; a valid starting time is generated randomly if it is invalid. Afterwards, some gene values can be altered in the staring time vector and workshop assignment vector to generate a new individual.

D. Chromosome Decoding

Decoding the chromosome is to convert an individual into a feasible schedule to calculate the objectives and constraints which represent the relative superiority of a chromosome. The group structure, the starting time and the workshop team of the operations can be obtained from each individual. Thereafter, the objective values can be calculated. When converting an individual to a schedule, it is possible that the processing times of two or more group operations assigned to the same workshop team are overlapping since the starting time of each group operation is fixed in the starting time vector. In this case, the principle of first-come-first-served is followed: the starting time and processing time of the earlier started group are not changed; the starting time of the later started group operation is not changed either; but the processing time of the later group operation is increased because the later group operation can be maintained only after the previous group operation is implemented. In our algorithm, the workshop abilities (components a workshop can maintain) are used as constraints to guarantee that a component will not be assigned to a workshop which cannot maintain it.

E. MOEAs

In the context of evolutionary multi-objective optimization (EMO), NSGA-III [12] and SMS-EMOA [13] are well-known MOEAs. SMS-EMOA belongs to indicator-based approaches and it uses the hypervolume indicator, which can capture both convergence and diversity, to guide the search. NSGA-III employs a widely-distributed set of reference points to efficiently promote the population diversity during the search and one reference direction could be associated with multiple solutions. Therefore, NSGA-III falls into the category of decomposition-based approaches. Diversity-Indicator based Multi-Objective Evolutionary Algorithm (DI-MOEA) [14] is a newly proposed MOEA for fast computation of evenly spread Pareto front approximations. It adopts a hybrid selection scheme: the (μ + μ) generational selection operator in the initial phase, to accelerate convergence until all solutions are non-dominated; the (μ + 1) steady state selection operator in the case that solutions are mutually non-dominated and the diversity is

the main selection criterion to achieve the uniformity. The Euclidean distance based geometric mean gap indicator is used as the diversity indicator to guide the search. Two variants of DI-MOEA, denoted as DI-1 and DI-2, exist, which employ crowding distance and the diversity indicator, respectively, as the second criteria in the generational (μ + μ) selection operator.

The proposed evolutionary algorithm framework has been combined with four MOEAs: NSGA-III, SMS-EMOA, DI-MOEA (DI-1 and DI-2) to solve our real-world vehicle fleet maintenance schedule optimization problem, and their performance is presented in the next section.

V. Experimental Results

A. Experimental Design

Three application instances with different size are generated and their parameters are listed in Table III. For example, the problem $P1$ includes 30 cars, each car consists of 4 components and 2 workshops are available; the two workshops have 3 and 4 repairing teams, and they can only maintain component o_1, o_2, o_3 and o_2, o_3, o_4 respectively. In order to make the results straightforward and better comparable, the processing time, cost as well as other variables are set to fixed values according to Table I; the due dates and previous repair dates are generated randomly in 100 days (negative values are used for the previous repair dates); vehicle demands are randomly generated for each problem in a way that the demand never exceeds the total number of cars in the fleet. All the problems and results are made available on http://moda.liacs.nl.

TABLE III. Problem parameters

Problem	#car	#comp	#ws	ws #car	ws #comp
$P1$	30	4	2	3	o_1, o_2, o_3
				4	o_2, o_3, o_4
$P2$	40	4	2	3	o_1, o_2, o_3
				4	o_2, o_3, o_4
$P3$	40	4	3	3	o_1, o_2, o_3
				4	o_2, o_3, o_4
				3	o_1, o_2, o_3, o_4

The application problems and algorithm have been implemented on the MOEA Framework (http://www.moeaframework.org/), which is a Java-based framework for multi-objective optimization. All the experiments are performed with a population size of 100 and a budget of 500000 fitness evaluations. Both the crossover and mutation probability is set to 1. For each problem, 30 optimization trials are performed with each algorithm. 10 cutting points are used in the multi-point crossover.

B. Computational Results

When analyzing the performance of multi-objective optimization algorithm, the empirical attainment function (EAF) is often used to visualize the attained parts of the objective space. The 50% attainment surface shows that half of all Pareto optimal solutions in the decision space will weakly dominate this surface and it is an estimator of what one would expect to achieve in 50% of runs [15]. Besides the average performance, the aggregate Pareto front approximation over 30 runs, i.e., the accumulated non-dominated solutions from 30 runs, has been used as another performance metric as well. Because extreme solutions are not preferable for our application problems and only one solution will be chosen to be deployed in workshops, we zoom in the Pareto front approximation and plot the 50% attainment surface and the aggregate Pareto front approximation on the knee regions. Since relatively loose vehicle demands are considered currently, all the vehicle demands have been satisfied in all the solutions on the Pareto front approximations we obtained from all algorithms. Therefore, we can only show the two-dimension plot.

Figure 7 shows the 50% attainment surface (upper row), aggregate Pareto front approximation of each algorithm (middle row) and aggregate Pareto front approximation of all algorithms (lower row) on three problem instances. When observing average performance from the 50% attainment surface, DI-2 behaves best for $P1$ and DI-1 behaves best for $P2$. For $P3$, DI-1 has an advantage in the region with lower cost and longer time and SMS-EMOA has an advantage in the region with shorter time and higher cost. When observing their respective aggregate Pareto front approximations, DI-1 has obvious advantages on all three instances; at the same time, DI-2 performs better on $P2$ and $P3$ than other algorithms. It is even clearer when we merge all optimal solutions from all algorithms. All optimal solutions on $P2$ and $P3$ are from DI-MOEA; most optimal solutions on $P1$ are from DI-1 and several are from SMS-EMOA.

We also examine the performance of these algorithms using the hypervolume indicator. Table IV shows the aggregate and median hypervolume across 30 runs on three problems. For each instance, the upper row is the aggregate hypervolume, the middle row is the median hypervolume and the lower row is the standard deviation; the best hypervolume value has been highlighted in bold. When calculating the hypervolume indicator, the reference point is used by the maximum extent of the population plus an offset. We can observe that SMS-EMOA, DI-2 and DI-1 performs best on $P1$, $P2$ and $P3$ respectively for the aggregate hypervolume; for the median hypervolume, SMS-EMOA performs best on $P1$, $P3$ and DI-1 performs best on $P2$.

TABLE IV. The aggregate and median Hypervolume (HV)

Algos / Probs		DI-1	DI-2	NSGA-III	SMS-EMOA
$P1$	Agg-HV	0.87983	0.91007	0.89932	**0.91633**
	M-HV	0.82728	0.83160	0.82506	**0.84137**
	std	0.0609	0.0506	0.0444	0.0563
$P2$	Agg-HV	0.89830	**0.98095**	0.93151	0.97677
	M-HV	**0.78950**	0.77907	0.72469	0.72647
	std	0.1649	0.1634	0.1259	0.1463
$P3$	Agg-HV	**0.88581**	0.88202	0.79875	0.87652
	M-HV	0.47185	0.51326	0.43115	**0.52816**
	std	0.2357	0.2244	0.2051	0.2424

For $P1$, if all the components are maintained exactly at their due dates, the total time and cost would be around 360

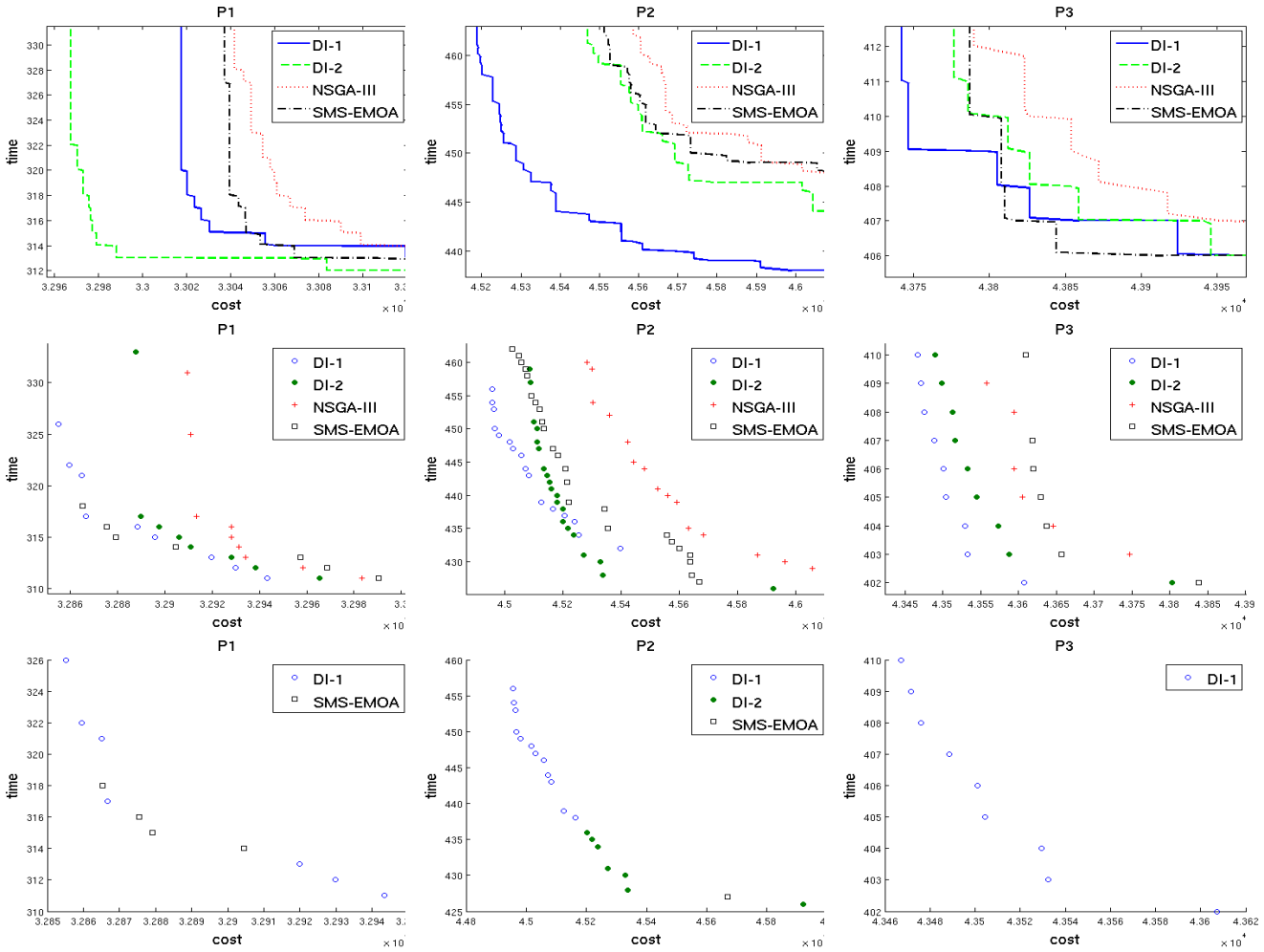

Fig. 7: The 50% attainment surface (upper row), aggregate Pareto front approximation of each algorithm (middle row) and aggregate Pareto front approximation of all algorithms (lower row) of three instances on cost(days) & time(€).

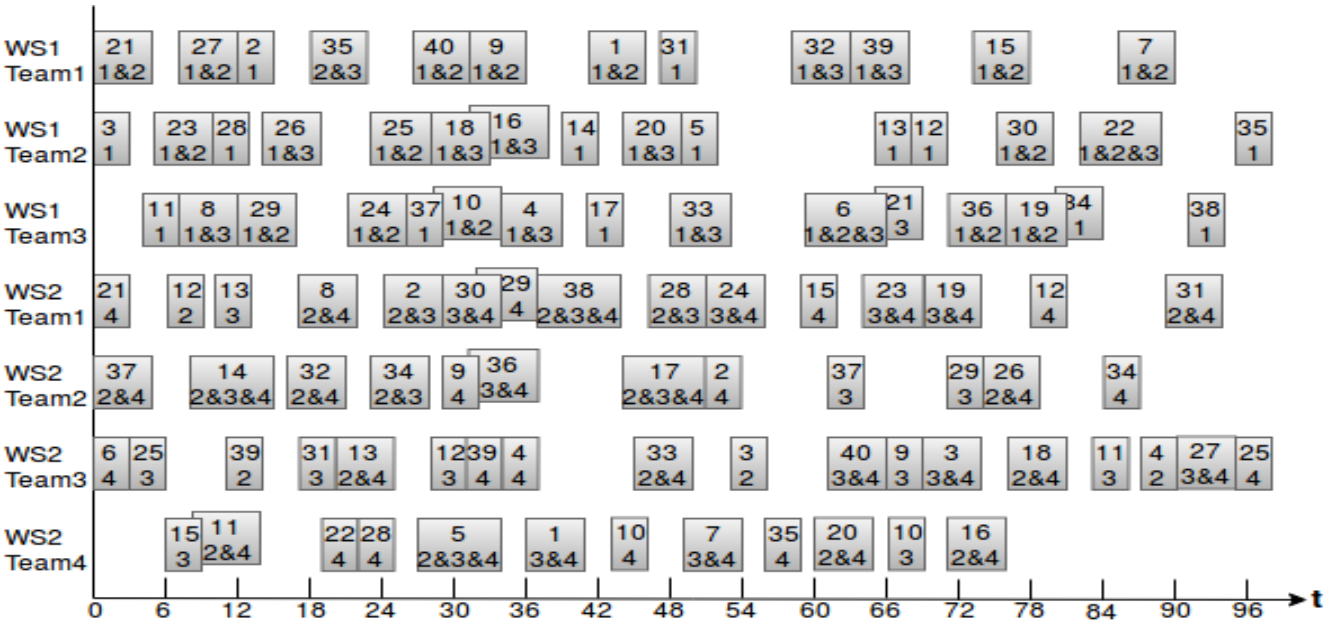

Fig. 8: A schedule on the Pareto front approximation of DI-2 on$P2$.

days and €36000, and vehicle demands cannot be guaranteed. Likewise, the total workload and cost would be around 480 days and €48000 for $P2$ and $P3$ when all the components are maintained exactly at their due dates. While, we can find many solutions with the workload less than 360 days and cost less than €36000 on the Pareto front approximations of all algorithms for $P1$. The same applies analogously for $P2$ and $P3$. Besides that the workload and cost of these solutions are already better than preforming the operations on the due dates, all the vehicle demands have been satisfied at the same time.

Figure 8 presents a schedule on the Pareto front approximation of DI-2 on $P2$, the two objectives (cost and time) of this solution are €45338 and 428 days, the customer demands are totally satisfied. Each item is a group operation, the number above is the car number and the number below is the component number. We can see that some components are maintained individually and some components of a car are grouped together for one workshop visit. There is no maintenance of component $c4$ in workshop $ws1$ because this workshop has no ability to maintain component $c4$, and the same for component $c1$ on $ws2$. It can also be observed that some operations are overlapping to some extend. For example, the first item on $Team4$ of $ws2$ is an operation of $c3$ on car 15, its maintenance starts from day 6 and ends on day 9 because the total processing time plus set-up time is three days. However, before the completion of this task, car 11 (for the operation of $c2$ and $c4$) is sent to the same team and has to wait for one day for maintenance. The reason why car 11 waits in the workshop instead of being sent to workshop one day later is that it takes more money if this car is on use for one more day due to the increase of failure probability of $c2$ or $c4$. There is no penalty cost for the wait of the car in the workshop because it has been "punished" by the increased processing time.

VI. Conclusions

The real-world problem of vehicle fleet maintenance scheduling is formulated. The penalty function is defined in order to deal with uncertainties in the due dates and to prevent too early or too late maintenance. A problem-specific multi-objective evolutionary algorithm framework is designed based on a grouping strategy. State-of-the-art MOEAs are applied with the algorithm framework to solve the problem and their behavior is investigated. Although DI-MOEA is used first time for a real application problem, its performance is comparable with and for some instances even better than other popular MOEAs such as NSGA-III and SMS-EMOA. DI-MOEA, especially DI-1, is the best both for the average performance and the aggregate Pareto front approximation. For the hypervolume indicator, DI-MOEA and SMS-EMOA are the best on all three instance.

The proposed algorithm framework can work for the generic application in various similar scenarios, for example, the aircraft maintenance, ship fleet maintenance, and so on. All the parameters can be flexibly adjusted based on the real application or by the decision maker. Right now, we only pay attention to the solutions on the knee region to find the final optimal schedule. In the future, one could search for solutions in the regions which are of real interest to the decision maker by running preference-based MOEAs, such as [16], [17]. In the real-world applications, it is desirable to generate schedules that are robust within a reasonable range of disruptions and uncertainties such as machine breakdowns and processing time variability. Therefore, dynamic elements will also be taken into account in the next step.

Acknowledgment

This work is part of the research programme Smart Industry SI2016 with project name CIMPLO and project number 15465, which is (partly) financed by the Netherlands Organisation for Scientific Research (NWO).

References

[1] https://www.acea.be/statistics/tag/category/size-distribution-of-vehicle-fleet retrieved on 17th of December 2018.
[2] Fleming, W.J., 2001. Overview of automotive sensors. IEEE sensors journal, 1(4), pp.296-308.
[3] Elattar, H.M., Elminir, H.K. and Riad, A.M., 2016. Prognostics: a literature review. Complex & Intelligent Systems, 2(2), pp.125-154.
[4] Jardine, A.K., Lin, D. and Banjevic, D., 2006. A review on machinery diagnostics and prognostics implementing condition-based maintenance. Mechanical systems and signal processing, 20(7), pp.1483-1510.
[5] Pierro, M.J. and Schneider, W.R., General Electric Co, 2001. Vehicle maintenance management system and method. U.S. Patent 6,301,531.
[6] Camci, F., 2009. System maintenance scheduling with prognostics information using genetic algorithm. IEEE Transactions on reliability, 58(3), pp.539-552.
[7] Bouvard, K., Artus, S., Brenguer, C. and Cocquempot, V., 2011. Condition-based dynamic maintenance operations planning & grouping. Application to commercial heavy vehicles. Reliability Engineering & System Safety, 96(6), pp.601-610.
[8] Chiang, T.C. and Lin, H.J., 2013. A simple and effective evolutionary algorithm for multiobjective flexible job shop scheduling. International Journal of Production Economics, 141(1), pp.87-98.
[9] Yuan, Y. and Xu, H., 2015. Multiobjective flexible job shop scheduling using memetic algorithms. IEEE Transactions on Automation Science and Engineering, 12(1), pp.336-353.
[10] Chaudhry, I.A. and Khan, A.A., 2016. A research survey: review of flexible job shop scheduling techniques. International Transactions in Operational Research, 23(3), pp.551-591.
[11] Özgüven, C., Özbakr, L. and Yavuz, Y., 2010. Mathematical models for job-shop scheduling problems with routing and process plan flexibility. Applied Mathematical Modelling, 34(6), pp.1539-1548.
[12] Deb, K. and Jain, H., 2014. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. IEEE Trans. Evolutionary Computation, 18(4), pp.577-601.
[13] Beume, N., Naujoks, B. and Emmerich, M., 2007. SMS-EMOA: Multiobjective selection based on dominated hypervolume. European Journal of Operational Research, 181(3), pp.1653-1669.
[14] Wang, Y., Emmerich, M., Deutz, A. and Bäck, T., 2019, March. Diversity-Indicator Based Multi-Objective Evolutionary Algorithm: DI-MOEA. In International Conference on Evolutionary Multi-Criterion Optimization (pp. 346-358). Springer, Cham.
[15] da Fonseca, V.G., Fonseca, C.M. and Hall, A.O., 2001, March. Inferential performance assessment of stochastic optimisers and the attainment function. In International Conference on Evolutionary Multi-Criterion Optimization (pp. 213-225). Springer, Berlin, Heidelberg.
[16] Wang, Y., Li, L., Yang, K. and Emmerich, M.T., 2017, June. A new approach to target region based multiobjective evolutionary algorithms. In Evolutionary Computation (CEC), 2017 IEEE Congress on (pp. 1757-1764). IEEE.
[17] Cheng, R., Jin, Y., Olhofer, M. and Sendhoff, B., 2016. A Reference Vector Guided Evolutionary Algorithm for Many-Objective Optimization. IEEE Trans. Evolutionary Computation, 20(5), pp.773-791.