Multi-objective co-operative co-evolutionary algorithm for minimizing carbon footprint and maximizing line efficiency in robotic assembly line systems

J. Mukund Nilakantan[1*], Zixiang Li[2], Qiuhua Tang[2], Peter Nielsen[1]

[1]Department of Mechanical and Manufacturing Engineering, Aalborg University, Denmark
Email: {mnj, peter}@m-tech.aau.dk

[2]Industrial Engineering Department, Wuhan University of Science and Technology, Wuhan 430081, Hubei, China
Email: zixiangli@126.com, tangqiuhua@wust.edu.cn

Abstract: Methods for reducing the carbon footprint is receiving increasing attention from industry as they work to create sustainable products. Assembly line systems are widely utilized to assemble different types of products and in recent years, robots have become extensively utilized, replacing manual labor. This paper focuses on minimizing the carbon footprint for robotic assembly line systems, a topic that has received limited attention in academia. This paper is primarily focused on developing a mathematical model to simultaneously minimize the total carbon footprint and maximize the efficiency of robotic assembly line systems. Due to the NP-hard nature of the considered problem, a multi-objective co-operative co-evolutionary (MOCC) algorithm is developed to solve it. Several improvements are applied to enhance the performance of the MOCC for obtaining a strong local search capacity and faster search speed. The performance of the proposed MOCC algorithm is compared with three other high-performing multi-objective methods. Computational and statistical results from the set of benchmark problems show that the proposed model can reduce the carbon footprint effectively. The proposed MOCC outperforms the other three methods by a significant margin as shown by utilizing one graphical and two quantitative Pareto compliant indicators.

Keywords: Robotic assembly line balancing; Carbon footprint; Multi-objective optimization; Co-evolutionary computation

1. Introduction

Assembly lines are flow-oriented systems of great importance in the automotive and consumer electronics industries. Robots have in recent years been widely applied in these types of systems, replacing manual labor (Gao et al., 2009). Robots are capable of operating 24 hours a day without worries of fatigue and can perform different tasks by re-programming. The effective utilization of robot assembly lines evolves into the need to solve the robotic assembly line balancing (RALB) problem, in which two sub-problems; task assignment and robot allocation, are addressed simultaneously.

Assembly line balancing is an important issue that must be addressed when considering the design of such systems. The literature study presented in this paper, shows that there is no research addressing the optimization of carbon footprint and line efficiency of robotic assembly line systems. This paper provides a method to simultaneously tackle the carbon footprint and line efficiency for a robotic assembly line system. The research contains two significant contributions to the field of robotic assembly line balancing. (1) A multi-objective generic model is developed to optimize the total carbon footprint and line efficiency. This is the first time the carbon footprint

is considered in robotic assembly line systems. (2) A multi-objective co-operative co-evolutionary algorithm (MOCC) is developed to simultaneously handle the task assignment and robot allocation. Multi-objective optimization is applied since the criteria of carbon footprint optimization and the line efficiency potentially conflict. MOCC is a new co-evolutionary method suitable for handling several sub-problems simultaneously and this algorithm suits this problem very well. MOCC is compared with three other multi-objective algorithms and a comprehensive study is carried out to test the superiority of the proposed MOCC.

The remainder of the paper is structured as follows. Section 2 presents a detailed literature review of the considered problem. Section 3 provides the problem assumptions and the mathematical model. Section 4 gives a detailed explanation of the proposed method along with a small-sized numerical example. Section 5 details the computational and statistical results. Finally, conclusions and future research avenues are presented in Section 6.

2. Literature review

Robotic assembly line balancing (RALB) problem was first introduced by Rubinovitz and Bukchin (1991). In Rubinovitz and Bukchin (1991) the allocation of the best available robot to the workstation to perform the allocated tasks is done based on the criteria of minimizing the number of workstations. Later, Rubinovitz et al. (1993) design and balance robotic assembly lines using branch-and-bound algorithm. Levitin et al. (2006) and Gao et al. (2009) solve RALB problems with the objective of minimizing cycle time through the use of genetic algorithms due to the NP-hard nature of the problem. Yoosefelahi et al. (2012) develop a multi-objective model and provide three multi-objective evolutionary strategies, while Dang et al. (2012) uses genetic algorithms to solve a multi-criteria problem of mobile robot scheduling. Most recently Nilakantan et al. (2015c) and Nilakantan and Ponnambalam (2016), utilize particle swarm optimization (PSO) algorithms and variants of PSO (Li et al., 2016a) to address different types of robotic assembly line balancing problems (one-sided, U-type and two-sided robotic assembly lines) with the objective of minimizing cycle time. Nilakantan et al. (2017) focuses on solving RALB problem with the objective of minimizing total assembly line cost and they utilized differential evolution algorithm to solve the problem. Most of the research is focused on RALB for single model assembly lines. However, Aghajani et al. (2014) consider the mixed-model two-sided RALB problem with a cycle time minimization criterion using a simulated annealing algorithm (Lee et al., 2001). The above mentioned literature mainly focused on RALB and different objectives analyzed over the years. The following paragraph discusses the literature related to energy consumption with respect to assembly line and manufacturing systems.

Research on automotive assembly by Fysikopoulos et al. (2012) report that energy costs contribute about 9-12% of the total manufacturing costs. Energy consumption cost is one of the major expenses for robotic assembly lines and one of the primary forms of energy used in the manufacturing sector is electricity. The manufacturing of electricity typically leads to emission of CO_2 which amounts to 20% of total emission in the

factories (Dai et al., 2013). Recently, Nilakantan et al. (2015a) investigate the energy consumption in straight robotic assembly lines and developed two models to minimize the cycle time and energy consumption. Due to the problem's NP-hard nature they utilized particle swarm optimization to solve the problem. Nilakantan et al. (2016) propose a set of new evolutionary algorithms for designing an energy efficient straight robotic assembly line. Nilakantan et al. (2015b) minimize the energy consumption of a U-shaped robotic assembly line. Li et al. (2016b) subsequently investigate the reduction of total energy consumption in two-sided robotic assembly lines and develop a multi-objective restarted simulated annealing algorithm to obtain Pareto solutions. Their results indicate that the optimization of line balancing and the minimization of energy consumption in some situations were conflicting. Recently, researchers propose metaheuristic approaches such as the genetic-simulated annealing algorithm (Tang and Dai, 2015) and the artificial bee colony algorithm to solve the scheduling problem of a flexible flow shop with the objective of reducing energy consumption. The reported results show that the proposed approaches could achieve on average a 10% reduction in the energy consumption when tested on small and medium sized problems.

Apart from the financial benefit of reduced energy consumption, the reduction also beneficially influences industry's impact on the environment. Yi et al. (2012) consider the carbon footprint in job-shop scheduling and a carbon footprint-aware model was developed to optimize makespan and carbon footprint. Liu (2014) develop a genetic algorithm to minimize total weighted tardiness and minimizing CO_2 emissions. Li et al. (2015) analyze the carbon emissions of CNC-based machining systems and consider the carbon footprint caused by cutting fluid, wear of cutting tools and material consumption. Lin et al. (2015) optimize the makespan and carbon footprint in turning processes using a multi-objective algorithm. Both the processing parameter optimization and flow-shop scheduling were addressed and the carbon footprints due to cutting fluids, disposal of worn tools and material consumption were considered. From literature it can be seen that minimal work has been reported to-date with respect to carbon footprint reduction and multi-objective optimization in robotic assembly line systems. Extensive searching has been done in Scopus database with the following keywords: robotic assembly line balancing, carbon footprint and mutli-objective and the search resulted in no relevant literature. The focus of this paper will be to develop a mathematical model and solution technique that simultaneously optimizes the carbon footprint and line efficiency.

3. Mathematical formulation

This section presents the mathematical model and the assumptions considered when solving the proposed problem.

3.1 Model assumptions

In robotic assembly lines, robots are allocated to each workstation and each performs a set of different assembly tasks. The balance of the assembly line and the allocation of the robots are two separate sub-problems that should be considered during the line balancing process. It is assumed that the carbon footprint is composed of energy

consumptions by a variety of different activities such as direct energy consumption, disposal of worn tools and material consumption (Lin et al., 2015). The disposal of worn tools and material consumption in robotic assembly lines can to some extent be regarded to be fixed as the same number of activities must be completed regardless of the number of robots used. This paper focuses on the carbon footprint caused by energy consumption. In this study, two types of energy consumption are considered. The first type of energy consumption is the energy consumed while performing the operation and the second type of energy consumption is the energy consumed while the robots are kept idle between operations. The following assumptions are similar to those presented in Gao et al. (2009) and Nilakantan et al. (2015c):

(1) Robots can be allocated to any workstation and can perform any task.
(2) The number of workstations is equal to the number of available robots and each robot is allocated to exactly one workstation.
(3) The operation times of tasks depend on the type of robots assigned and the operation times for a task vary depending on the robot completing the operation.
(4) A task can be allocated only when the cycle time and the precedence constraint(s) are satisfied.
(5) Only one kind of product is assembled in straight assembly lines and parallel workstations are not considered.
(6) Setup times, work-in-process inventory and material handling are negligible.
(7) The carbon footprint is caused by the power consumptions of robots and the carbon footprint by other activities is negligible.
(8) The power consumption consists of operation power consumption and standby power consumption, where the standby power consumption is the power consumption during the intervals between performing tasks.
(9) This planning horizon is not considered and the maintenance operations are negligible in this study.
(10) The energy consumed during the maintenance/disposal stage is not considered.

3.2 Notations

The notations in the mathematical formulation are introduced as follows.

▪ *Indices:*
i,j: Index of tasks
s: Index of stations
r: Index of robots

▪ *Parameters:*
Nt: Total number of tasks
Ns: Total number of workstations
Nr: Total number of robots
OPC_r: Operation power consumption of the robot r per time unit
SPC_r: Standby power consumption of the robot r per time unit

ECF_{elec}: The average electricity carbon footprint of the main power grids (ECF_{elec}=0.5488 kg CO_2/ kWh (Lin et al., 2015))
t_{ir}: Operation time of task i by robot r
$Pre(i)$: Set of immediate predecessors of the task i

▪ *Decision variables:*
CFE: Carbon footprint by energy consumption.
LE: Line efficiency.
TEC: Total energy consumption.
EC_s: Energy consumption on workstation s.
OEC_s: Operation energy consumption on workstation s.
SEC_s: Standby energy consumption on workstation s.
CT: Cycle time.
x_{is}: 1, if task i is assigned to workstation s; 0, otherwise.
y_{rs}: 1, if robot r is allocated to workstation s; 0, otherwise.

3.3 Mathematical model for RALB

The mathematical formulation for the RALB problem is developed with the aim of optimizing two objectives. The first objective is to maximize the line efficiency and the second objective is to minimize the carbon footprint caused by the energy consumption. The detailed mathematical formulation is presented as follows. To a common format, maximizing LE in expression (1) is replaced with minimizing 1-LE so that both objectives are minimized. Although both these objectives address reducing idle times, they might conflict in some occasions due to the diverse operations caused by different robot allocations.

$$\max LE = \sum_{s=1}^{N_s}\sum_{r=1}^{N_r}\sum_{i=1}^{N_t} t_{ir}.x_{is}.y_{rs}\ /(CT.NS) \tag{1}$$

$$\min\ CFE = ECF_{elec}.\sum_{s=1}^{N_s} EC_s \tag{2}$$

$$EC_S = OEC_S + SEC_S \tag{3}$$

$$OEC_s = \sum_{r=1}^{Nr}\sum_{i=1}^{Nt} OPC_r \cdot t_{ir} \cdot x_{is} \cdot y_{rs} \tag{4}$$

$$SECs = CT.(\sum_{r=1}^{Nr} SPC_r.y_{rs}) - (\sum_{r=1}^{Nr}\sum_{i=1}^{Nt} SPC_r.t_{ir}.x_{is}.y_{rs}) \tag{5}$$

$$CT = \max_{1\le s\le N_s}\{\sum_{r=1}^{r=N_r}\sum_{i=1}^{i=N_t} t_{ir}.x_{is}.y_{rs}\} \tag{6}$$

$$\sum_{s=1}^{N_s} s.x_{is} - \sum_{s=1}^{Ns} s.x_{js} \le 0\ \ \forall i \in \text{Pre } (j); j \tag{7}$$

$$\sum_{s=1}^{s=N_s} x_{is} = 1 \quad \forall i \tag{8}$$

$$\sum_{r=1}^{r=N_r} y_{rs} = 1 \quad \forall s \tag{9}$$

$$\sum_{s=1}^{s=N_s} y_{rs} = 1 \quad \forall r \tag{10}$$

Objective (1) maximizes the line efficiency of the robotic assembly line. Objective (2) minimizes the total carbon footprint by energy consumptions on all the workstations. Equation (3) indicates that the energy consumption on a workstation consists of operation energy consumption and standby energy consumption. Equation (4) calculates the energy consumption on a station, while constraint (5) calculates the standby energy consumption at a station. Equation (6) indicates that the cycle time is larger than the largest finishing times of tasks on all stations. Equation (7) deals with precedence relationship which ensures that the predecessor of task i must be allocated before task j. Equation (8) guarantees that each task has to be allocated to a workstation. Equation (9) indicates that each workstation is allocated a robot and Equation (10) ensures that each robot is allocated to a workstation.

Note that this mathematical model considers the carbon footprint caused by the total energy consumption. The total energy consumed is the sum of all the energy consumptions on all workstations as reported in Nilakantan et al. (2015a). In addition, energy consumption on each workstation is divided into two modes, operation mode and standby mode. Standby mode is the intervals between operations (idle time) and the standby power consumption of a robot per time unit can be considered to be 10% of the operation power (Bertoldi et al., 2002).

4. Multi-objective co-evolutionary algorithm for RALB problem

Since the problem consists of two sub-problems and two optimization criteria, this research utilizes a multi-objective co-operative evolutionary algorithm (MOCC). This algorithm shows promising and competitive results for some multi-objective optimization problems (Zhao et al., 2014). The following sections explain the implementation steps of this algorithm.

4.1 Procedure of MOCC

The proposed MOCC has two sub-populations which optimize the task assignment and robot allocation. The rationale of the proposed algorithm is quite simple: a co-evolutionary greedy multi-objective strategy is iteratively applied over a set of non-dominated solutions. An extending operator and a restart mechanism are embedded to emphasize exploration. The procedure of co-evolution is depicted in Figure 1.

The proposed MOCC starts with an empty external archive (A), an empty current archive (CA) and an initialization procedure. One hundred random individuals containing two representatives are obtained in the initialization procedure. These are

used to obtain the initial current archive. One individual is selected from the current archive as the best individual (*BI*), which contains two representatives. Following steps are performed in a cycle until a termination criterion is satisfied. First, the new population *Pop* (*p*) is updated with *Popsize*-1 neighbor solutions of *BI* and one neighbor solution of a randomly selected one from *CA*. Second, a new solution *CP*(*n*) is generated by combining the other representative of *BI*. The best representative from the first *Popsize*-1 is utilized to replace the current *BI* if new non-dominated solutions can be obtained by them. Subsequently, the *BI* is replaced with a solution from *CA* based on crowding distance if a new non-dominated solution has not been achieved for *NI* times' iteration. Optimally a restart mechanism is applied when the current archive cannot obtain better results and this paper introduces two new types of restart methods. If the external archive is updated with the current archive, a set of new neighbor solutions of the individuals in the external archive are generated to obtain the new current archive. If the external archive cannot be updated, a set of randomly generated solutions are utilized to obtain the new current archive.

Note that the proposed MOCC differs from the one proposed by Zhao et al. (2014) by a significant margin in co-evaluation method and extending operator. The main idea behind these modifications is to enhance intensification that will be further explained later. Especially, the proposed MOCC introduces a restart mechanism to escape from local optima. Detailed explanations of these modifications are introduced in the following subsections.

4.2 Solution representation and initialization

The task assignment vector and robot allocation vector are adopted for encoding and decoding. The task assignment vector is a $1\times Nt$ vector and each element corresponds to a workstation. Suppose that the i^{th} element is s, task i is assigned to the s^{th} workstation. The robot allocation vector is a $1\times Nr$ vector and each element also corresponds to a robot. If the s^{th} element is r, robot r is allocated to workstation s. An example of the encoding and decoding procedure is depicted in Figure 2. As shown in figure the elements in the first position and the second position are 1 and 2 and, task 1 and 2 are assigned to workstation 1 and 2. The robot sequence is 3, 2, 1 and 4, and due to which robot 3, robot 2, robot 1 and robot 4 are allocated to workstation 1, workstation 2, workstation 3 and workstation 4. The task to be assigned to a station is determined by the precedence relationships. A task is allocated only when all predecessors have been allocated. For instance, the predecessor of task 3 is task 1, due to which task 1 is assigned first.

```
Algorithm MOCCA for multi-objective RALB problem
Input: Popsize (subpopulation size), RALB data
Output: A (External archive)
Begin:
  A:=Ø; CA=Ø;
  For n=1, 2 to 100 do     //Initialization
     CP(n)=nth randomly initialized representatives;
     CA=NondominatedSet(CP U CA);
  BI= A complete solution selected from current archive based on crowing distance;
  While (termination criterion is not satisfied) do
     For p=1 to 2 do       //Evaluation
        Replace Pop(p) with (Popsize-1) neighbor solutions of BI and one neighbor
        solution of a random selected one from current archive     //Co-evolution
        For n=1, 2 to Popsize do
           CP(n) is a complete solution which combines the representative n in subpopulation
           Pop(p) and other representative from BI
           CA= NondominatedSet(CP(n) U CA);   //Updating the current archive
        If (New non-dominated solution is obtained for the first Popsize-1 individuals)
           BI=BestIndividual from CP(1), CP(2) to CP(Popsize-1) ;
     If (New non-dominated solution has not been found for NI times' iteration)
        // Extending operator
        BI= A complete solution selected from current archive based on crowing distance;
     If (All the individual has been selected to replace BI at least for once)
        //Restart mechanism
        A=NondominatedSet(A U CA); Empty (CA) ;
        If (External archive is updated by CA)
           For n=1, 2 to Size(A) do
              CP(n)=nth neighbor representatives in external archive;
        Else
           For n=1, 2 to 100 do
              CP(n)=nth randomly initialized representatives;
        CA=NondominatedSet(CP U CA);
        BI= A complete solution from current external archive based on crowing distance;
End
```

Figure 1 The co-evolution procedure of MOCC

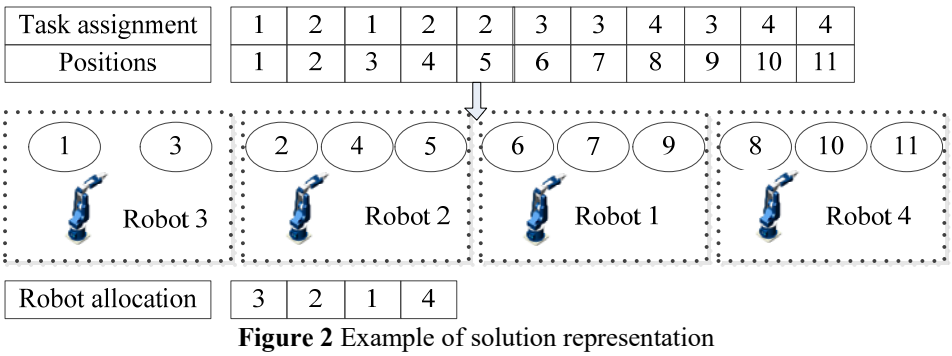


Figure 2 Example of solution representation

In the initialization process, the task assignment vector is randomly generated. A value between [1, *Ns*] is generated randomly for each position and an individual is generated after generating all values for all positions. The robot allocation is also randomly generated. First, a random value between [1, *Nr*] is obtained, then a second

value is obtained from the remaining robots and finally the remaining robot is allocated to the last workstation. The robot allocation vector does not affect the feasibility of the solutions whereas the task assignment vector may result in infeasible solutions due to the precedence relationships. This paper develops a simple repair procedure to create an initial feasible solution. For each task i, if the predecessor j of task i is allocated to the latter workstation, then the corresponding values of the two tasks in the tasks assignment vector are exchanged. This repair procedure terminates only when the predecessor of each task is allocated to the former or same workstation. After executing this procedure, the precedence constraint is satisfied, and then the tasks are allocated in sequence on a workstation based on the precedence constraint.

4.3 Population update and solution evaluation

As for the population update, this paper implements a big adjustment to the current method as found in Zhao et al. (2014). The sub-population *Pop*(*p*) is updated by *Popsize*-1 neighbor solutions of *BI* and one neighbor solution of a randomly selected solution from *CA*. In general, the new sub-population is updated using selection, crossover operator and mutation operator on the current sub-population. The proposed modification is carried out to achieve computational speed benefits and provides a strong local search on *BI*. In the preliminary experiments, the new population update is faster than the original one used in (Zhao et al., 2014). The new sub-population also includes one neighbor solution of a randomly selected solution from *CA* to emphasize exploration. To obtain a neighbor solution, two types of neighborhood structures are applied for the task assignment and robot allocation vectors. For the task assignment vector the alteration operator and swap operator are applied and randomly selected. The procedure is depicted in Figure 3. In the alteration operator, a position is selected and the workstation of this position is changed to another workstation. In the swap operator, the values on two different positions are exchanged. As for the robot allocation vector, the insert operator and swap operator are applied and randomly selected.

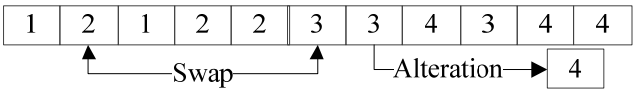


Figure 3 Mutation operator for task assignment

To evaluate an individual in a sub-population, it is necessary to take one individual from the other sub-population to obtain a completed solution. This paper combines the evaluated individual with the other representative from *B*I. This new evaluation procedure differs from the one utilized in (Zhao et al., 2014) where both the best individual and a random individual from other sub-population are utilized. This research does not utilize a random individual since the combined solutions with random individuals in the preliminary experiments conducted in this study always have poor performances. This gives poor computational performance with regards to convergence towards a high quality solution. Subsequently, if a solution is not dominated by any solution in the current archive, this solution is a new non-dominated solution. This new non-dominated solution is added into the current archive and original solutions which

are dominated by this solution are removed from the current archive.

After obtaining all the individuals for each sub-population, the fitness of an individual is determined by checking whether the solutions in the external archive dominate it. If all the new individuals are dominated, the *BI* remains unchanged. On the contrary, if at least one individual among the former *Popsize*-1 individuals is not dominated by the current archive, the *BI* is replaced with first one of the non-dominated individuals. This modification converts the update of *BI* into a greedy process and guarantees that the search is carried out only near non-dominated solutions. Notice that the *BI* is only updated when at least one of the former *Popsize*-1 individuals is non-dominated. The last individual is a neighbor individual of a randomly selected solution from *CA*. If the *BI* is replaced with the last individual, the new *BI* has a very small relationship with the original *BI* and the current *BI* will be abandoned without going through an exhaustive local search.

4.4 Extending operator and restart mechanism

When utilizing the greedy process on *BI* the algorithm easily gets trapped in local optima is one of the drawbacks. For this reason, when a new non-dominated solution has not been achieved within *NI* iterations, the extending operator takes effect and the current *BI* is replaced with a solution from *CA*. In this research, an adjustment is conducted on the original extending operator by replacing the *BI* with a selected solution from *CA* rather than replacing a random individual in the sub-population as done in Zhao et al. (2014). This modified extending operator along with the population update guarantees the strong local search capacity of the individuals in *CA*.

Another problem is to select an individual in *CA*. In general, there are many solutions in *CA* and selecting correctly has a non-trivial impact on the performance. In this paper, the extending operator is applied to guide the sub-population to search the regions that are not explored enough and the least crowded individual in the current archive is usually selected to replace *BI*. To achieve this, a modified crowding distance mechanism is developed, as plotted in Figure 4, to select an individual by taking the selection times of the solutions into account. The boundary solutions are set to the maximum of the distance of all other individuals rather than a big number. The selection time is also adopted and it ensures that the distances of the selected solutions become smaller. These two modifications to the crowing distance mechanism ensure that the isolated solutions that are less selected have a large possibility of being selected. It should be noted that the crowding distance in this selection mechanism is different from the one used by Li et al. (2016b). This modification aims at reducing the crowding distance of selected individuals at a faster rate.

Modified crowding distance mechanism

% A is the external Pareto-optional set, *Obj* means objective, *dist* indicates distance of individual *n*, *Setsize*:= |*A*|, *SelectCounter* means the number of being selected

For all objectives *m*

Sort A using the current objective *m*

$f_m^{\max}$:=maximum value for the objective *m*

$f_m^{\min}$:=minimum value for the objective *m*

For all *n*:=2,⋯,*Setsize*-1

$A[n]_{dist} := A[n]_{dist} + \left(A[n+1]_{Obj_i} - A[n-1]_{Obj_i} \right) \Big/ \left(f_m^{\max} - f_m^{\min} \right)$;

End for

$A[1]_{dist} := A[1]_{dist} + \max_{n=2,\cdots,Setsize-1} A[n]_{dist}$;

$A[Setsize]_{dist} := A[Setsize]_{dist} + \max_{n=2,\cdots,Setsize-1} A[n]_{dist}$;

End for

For all *n*:=1,⋯,*Setsize*

$A[n]_{dist} := A[n]_{dist} := A[n]_{dist} \cdot \text{pow}(0.5, A[n]_{SelectCounter})$;

End for

Select the individual with largest distance;

Figure 4 Modified crowding distance mechanism

If all the individuals in the current archive have been selected to replace *BI* at least once, there is a large possibility that the current archive cannot be updated with a small adjustment. A restart mechanism is very necessary and this paper develops two types of restart methods based on whether the external archive can be updated with the current archive or not. If so, there is still a potential that the external archive can be improved and the neighbor solutions of the external archive are generated to obtain a new current archive. This method preserves the high quality of the current archive while preserving the diversity. In this research, both task assignment and robot allocation of the individual in the external archive is modified using one of the neighborhood structures to obtain neighbor solutions. If the external archive cannot be updated, the method used in this paper completely restarts the algorithm with a set of randomly generated solutions.

4.5 Numerical example

This section illustrates the process followed for obtaining the objectives by solving a small-sized problem. This problem contains 11 tasks and is completed by 4 robots on 4 workstations. Table 1 shows the detailed operation times of tasks by each robot. It can be observed that the tasks in some cases have different operation times by different robots. The assignment and the robot allocation are depicted in Figure 5 and from this, the workstation time on each workstation can be obtained. For instance, the operation times of task 1 and task 5 by robot 3 are 64 and 48, and the operation time on workstation 1 is 64+48=112.

Table 1 Operation times of 11 tasks by 4 robots

Task	Robot 1	Robot 2	Robot 3	Robot 4
1	135	65	64	46
2	58	27	30	21
3	37	30	42	26
4	62	40	68	62
5	92	37	48	38
6	100	61	99	27
7	156	90	73	36
8	147	37	108	47
9	122	28	59	29
10	66	30	72	86
11	70	29	35	21

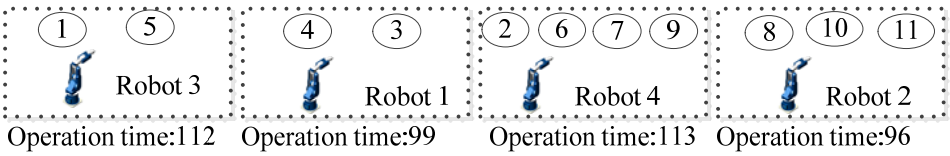


Figure 5 Example of task assignment and robot allocation

Suppose that the energy consumptions of the robots are: 0.25, 0.4, 0.3 and 0.35. Then the carbon footprint caused by energy consumption and the line efficiency is calculated in Table 2. The robots follow the power consumption scheme previously listed in Section 2.3. Operation energy consumption can be achieved with the product of operation time and operation power consumption per time unit using Equation (4). The standby power consumption is calculated with the same method using Equation (5). The total energy consumption is the sum of the operation power consumption and standby power consumption on all workstations using Equation (2), namely 137.36 units. The carbon footprint by energy consumption and line efficiency are achieved with Equation (2) and Equation (1).

Table 2 Carbon footprint and line efficiency evaluation

Workstations	1	2	3	4
Cycle time	113	113	113	113
Robot allocation on workstations	3	1	4	2
Operation time on workstation *s*	112	99	113	96
Standby time on workstation *s*	1	14	0	17
Operation power consumption per time unit	0.3	0.25	0.35	0.4
Standby power consumption per time unit	0.03	0.025	0.035	0.04
Operation energy consumption on workstation *s*	33.6	24.75	39.55	38.4
Standby energy consumption on workstation *s*	0.03	0.35	0	0.68
Total energy consumption	137.36			
Carbon footprint by energy consumption	75.383168			
Line efficiency	0.92920354			

5. Computational study

A series of computational experiments are carried out to test the performance of the proposed multi-objective model and the proposed MOCC. Details of the benchmark problems and the evaluation methodologies are introduced and the parameters of MOCC are presented. Subsequently, the MOCC is compared with adaption of three multi-objective algorithms to test the performance of the proposed MOCC.

5.1 Benchmark problem and adaption of multi-objective algorithms

The datasets proposed by Gao et al. (2009) for robotic assembly line balancing

problems is adopted in this research to evaluate the multi-objective model and algorithms. There are 32 test problems with 8 precedence graphs. Operation power consumption per time unit is taken from Nilakantan et al. (2015a) and the standby power consumption per time unit is 10% of the operation power consumption (Bertoldi et al., 2002). The average electricity carbon footprint of the main power grids is 0.5488kg CO_2/ kWh (Lin et al., 2015). The tested problems range from small-sized problem with 25 tasks and large-sized problem with 297 tasks. In this study, the datasets are divided into two groups: small-sized problems with 25 to 53 tasks and large-sized problem with 70 tasks to 297 tasks.

To evaluate the proposed MOCC, this paper re-implements three multi-objective algorithms. The first is the fast elitist non-dominated sorting genetic algorithm (NSGA-II) (Deb et al., 2002). NSGA-II is one of the most commonly used algorithms for solving multi-objective models and is used as a basic benchmark algorithm for result comparison by many researchers (Saif et al., 2014). The second algorithm is the restarted simulated annealing algorithm (RSA) as proposed by Li et al. (2016b). The RSA has proven to outperform NSGA-II in both convergence and spread criteria. It should be noted that although there are many multi-objective algorithms, these are the only two multi-objective algorithms that have been applied to solve RALB problems under the condition of utilizing two vectors for encoding. In addition, in this paper another multi-objective artificial bee colony algorithm (ABC) proposed by Saif et al. (2014) is also selected for the comparative study. ABC is included as it also optimizes two vectors simultaneously and this mechanism suits the problem considered in this research. ABC utilizes the new encoding and decoding in Section 3.2. The pseudocodes of the algorithms are not presented due to space constraints. Pseudo codes along with codes in C++ programming language are available upon request to the authors.

Another important issue for algorithm comparison is the choice of a proper termination criterion. This research utilizes elapsed CPU time as the termination criteria which assign more time to larger problem instances. This elapsed CPU time is expressed with $Nt \times Nt \times \tau$ milliseconds, where τ an input parameter. This termination criterion has previously been applied for solving assembly line balancing problems (Li et al., 2016c). The expression ensures that the computational time increases as the size of problems increases. In this paper, τ is tested on four levels where τ is set to be equal to 10, 20, 30 and 40. Four termination criteria provide the information on the performance of algorithms from short computational time to very large computational time and also avoid prejudiced comparison to increase the soundness of the experimental conclusions. All the experiments are carried out on a set of personal computers with Intel(R) Core2(TM) CPU 2.33GHZ and 3.036 GB RMA and all the algorithms are coded in C++ programming language on the Microsoft Visual Studio 2012 platform.

5.2 Performance evaluation methodologies

The evaluation of single-objective algorithms is straightforward whereas the evaluation of the Pareto archives is rather complicated. The easiest comparison happens when an archive *A* is dominated by an archive *B*, but this situation rarely happens in real applications. When this situation does not occur, the determination of which

archive is better is a big challenge. According to Ciavotta et al. (2013), there is no unambiguous way to determine which frontier is better. In this paper, two Pareto compliant quantitative performance indicators are adopted to measure the performance of these algorithms: hyper volume ratio (*HVR*), and Unary Epsilon Indicator (I_ε^1). It should be noted that other indicator, such as convergence of the Pareto-optimal set and spread metric, are not Pareto compliant (Ciavotta et al., 2013) and might give wrong and misleading results. The two Pareto compliant indicators are explained as follows. *HVR* (Zhao et al., 2014) is the ratio of the hyper volume of the obtained frontier (*S*) and that of the optimal or near-optimal Pareto front (*P*), which is calculated with the following Equation (11).

$$HVR = \frac{volume(U_{i=1}^{size(S)} \quad x_i)}{volume(U_{j=1}^{size(P)} \quad x_j)} \tag{11}$$

In *HVR*, the hypercube is constructed with a reference point *W* and the frontier curve and the solutions in the frontier curve are the diagonal corners of the constructed hypercube. The reference point *W* can be achieved by combining the worst objective values for all objectives. Zitzler et al. (2001) indicates that *HVR* can be considered to be the most appropriate scalar indicator since it combines both the convergence to the true frontier and the spread of the solutions. A value close to 1.0 indicates a better approximation to the true frontier.

The Unary Epsilon Indicator (I_ε^1) is also a Pareto compliant indicator, which measures the minimum distance between a given frontier and the optimal or near-optimal frontier. This indicator has been successfully applied by among others Ciavotta et al. (2013). In this indicator, the objectives are normalized and they are additionally transferred by adding one unit to avoid division by zero errors in the calculation process. Formally, this indicator is calculated with Equation (12) as follows:

$$I_\varepsilon^1 = I_\varepsilon(S,P) = max_{x^2}\, min_{x^1}\, max_j \frac{f_j(x^1)}{f_j(x^2)} \tag{12}$$

A I_ε^1 value close to 1.0 indicates that the obtained frontier is close to the true frontier whereas a value close to 2.0 indicates a large distance to the true frontier.

One possible drawback of these indicators is that they cannot provide information about the spatial performance of the compared algorithms e.g. which algorithm performs better in a region of the objective space. This research proposes to utilize a probabilistic measure called the Attainment Function to show the spatial behavior of multi-objective algorithms. The Attainment Function was first developed by da Fonseca et al. (2001) and this method has been successfully applied to solve multi-objective flow shop problems (Ciavotta et al., 2013). Let $x_l \in \mathbb{R}^d$ be an arbitrary point in the *d*-objective solution space and $\Gamma = \{x_l \in \mathbb{R}^d, l = 1,2,\dots,|P_a|\}$ be a non-dominated solution set by an algorithm in a single run. The Attainment Function is calculated with Equation (13). Equation (13) describes the probability that the algorithm α generates an approximated Pareto front P_α in which at least one element weakly dominates an arbitrary point in a single run. Since the algorithms are stochastic, it is impossible to obtain accurate values and it is empirically approximated by the outcomes of running the algorithms repeatedly. This method is expressed with Equations (14-15), called the Empirical Attainment Function (EAF). In Equation (14), Γ_h is the Pareto set in h'th

repetition of the algorithm and *Nh* is the total number of repetitions of the algorithm independently. Equations (14-15) can obtain the probability of obtaining a Pareto front in which at least one solution dominates the point α in a single run. To compare the performance of several methods, the difference between two EAFs denoted as Diff-EAF is defined in Equation (16) (López-Ibánez et al., 2006). This expression denotes the probability of ω only being dominated by a Pareto front of algorithm α and not of algorithm β. The value of Diff-EAF can be negative or positive. If the value is positive, algorithm α has more opportunities for obtaining a Pareto front in which at least one solution dominates β. If the value is negative the reverse is true.

$$AF_{\alpha}(\omega) = P(\exists x_1 \in 1: x_1 \trianglelefteq \omega) \tag{13}$$

$$EAF_{\alpha}(\omega) = \frac{1}{N_h}\sum_{h=1}^{Nh} I(\Gamma_h \trianglelefteq \omega) \tag{14}$$

$$I(\Gamma_h \trianglelefteq \omega) = \begin{cases} 1 & if \Gamma_h \trianglelefteq \omega \\ 0 & otherwise \end{cases} \tag{15}$$

$$Diff - EAF_{(\alpha,\beta)}(\omega) = \frac{1}{Nh}\sum_{h=1}^{Nh}[I(\Gamma_h^{\alpha} \trianglelefteq \omega) - I\left(\Gamma_h^{\beta} \trianglelefteq \omega\right)] \tag{16}$$

5.3 Parameter calibration of proposed methods

In the following the best parameter configuration for the MOCC and other multi-objective methods is selected. The calibration process of MOCC is illustrated as an example, in which the population size (*Popsize*) and the number of iterations (NI) need to be determined prior to executing the extending operator presented in Section 4.4. This paper proposes a full factorial design of experiments similar to the one proposed in Li et al. (2016c) to investigate the parameter setting. To reduce the calibration experiments to manageable levels, the ranges of the parameters are first determined by fixing the remaining parameters, and only several best parameters are selected for further calibration. In this paper, the *Popsize* is tested at five levels and *NI* is tested for four levels and the ranges for the two parameters are presented in Table 2.

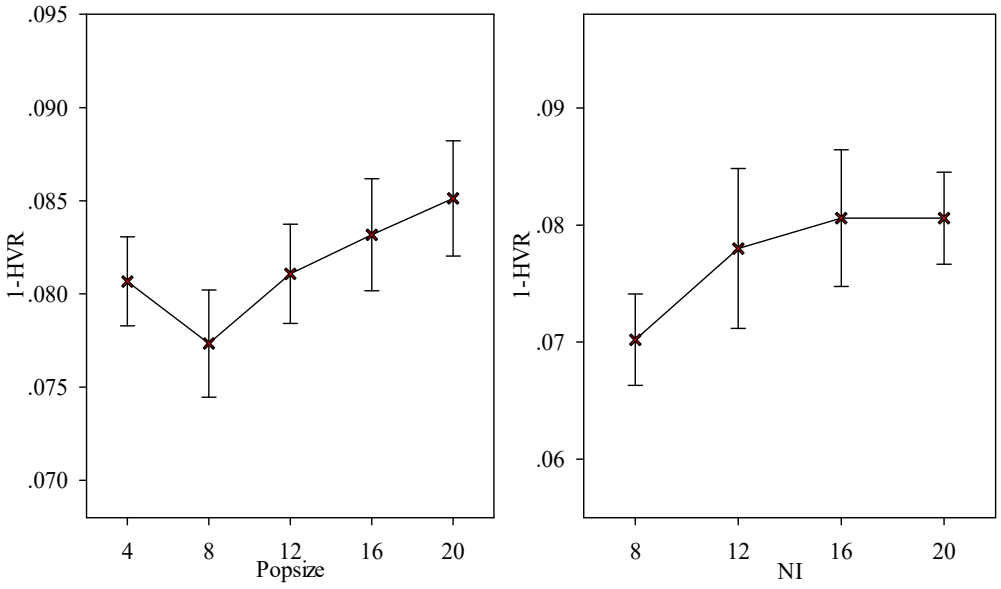
Ten cases of P70 (problem with 70 tasks) are selected for the calibration and this case is solved ten times. The termination criterion is set to be equal to an elapsed CPU time of $Nt \times Nt \times 10$ milliseconds. *1-HVR* is selected as the response variable since *HVR* can be considered to the most appropriate scalar indicator as stated by Zitzler et al. (2001). An analysis of variance (ANOVA) technique (Montgomery, 2008) is applied to analyze the results. To apply ANOVA, the following three hypotheses should be fulfilled: independence of residuals, homogeneity of variance and normality of the residuals. As in the research presented in Tang et al. (2016) the ANOVA test is carried out and the detailed ANOVA table is omitted due to space constraints. Instead, this section illustrates the means plots with Tukey HSD intervals for two parameters as depicted in Figure 6. Clearly, the combination of 8 for *Popsize* and 8 for *NI* obtains the peak performance and these two values are selected for the following computational experiments.

The parameters for the other algorithms are calibrated in the same manner. The parameter configuration utilized in this paper is presented in Table 3. It might be argued that the proposed algorithms can be further improved by consecutive rounds of tuning

of a few significant parameters. For instance, the current step for population size of NSGA-II is 60 and smaller steps might further improve the performance. This consecutive round calibration is not carried out since performance differences in preliminary experiments are too small to be relevant. Another issue is whether the calibrated parameters can be shared by all the benchmark problems. In the conducted tests, the parameters of large-sized problems are similar, but they are different from that of small-sized problems. Since this calibration process only considers large-sized problems, another calibration is necessary for small-sized problems. It is to be noted that parameter calibration is just a fine tuning process and algorithms might perform similarly even after calibration.

5.4 Computational result analysis

In the following section, the proposed multi-objective model is tested and a comparison campaign among the tested methods is shown. To test the rationality of the multi-objective model it is compared with two single-objective models to minimize 1-line efficiency and total carbon footprint. These three models are embedded into the simulated annealing algorithms and there are three simulated annealing algorithms: RSA with two objectives, SA-1 with the objective of minimizing the 1- line efficiency and SA-2 with the objective of minimizing the total carbon footprint.



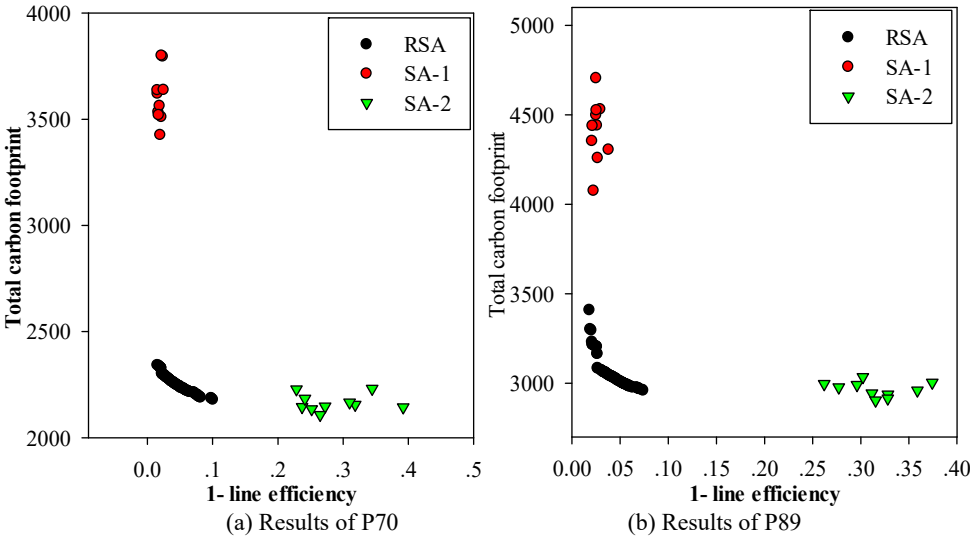
(a) Means plot for Popsize (b) Means plot for HI

Figure 6 Means plot and Tukey HSD intervals at the 95% confidence level for two parameters

Table 3 Parameter values of tested algorithms

Parameters	Range	Value
MOCC algorithm		
Population size	4, 8, 12, 16, 20	8
Number of iterations before executing extending operator	8, 12, 16, 20	8
NSGA-II algorithm		
Population size	60, 120, 180, 240, 300	60
Selection type	Random, tournament	Random
Crossover rate	0.1, 0.2, 0.3, 0.4, 0.5	0.4
Mutation rate	1- Crossover rate	0.6
ABC algorithm		
Population size	60, 120, 180, 240, 300	60
RSA algorithm		
Initial temperature	0.5, 1.0	1.0
Cooling rate	0.9, 0.95, 0.98	0.9
Number of iterations before a temperature change	500, 1000, 1500	500
Number of moves before restart	100, 200, 300	100

To demonstrate the performances of the three algorithms, the results from P70 with 14 workstations and P89 with 16 workstations is shown. Each case is solved ten times under the termination criterion of an elapsed CPU time of $Nt \times Nt \times 20$ milliseconds. Figure 7 (a-b) plots the Pareto archive by RSA and the twenty solutions obtained using by SA-1 and SA-2 for two selected problems. The Pareto archive depicted only contains the non-dominated solutions for better visualization. In Figure 7 (a), SA1 is capable of obtaining better results regarding line efficiency than SA2 with the cost of obtaining a large carbon footprint of about 3400 units. On the other hand, SA2 obtains much better result regarding carbon footprint, but has poor performance regarding line efficiency. The proposed RSA inherits the advantages of SA1 and SA2 by obtaining a Pareto front. It is observed that the majority of the results obtained by SA-1 and SA-2 are dominated by the Pareto archive by RSA. This small comparison indicates that the multi-objective model is reasonable, and it is able to reduce the carbon footprint effectively.



(a) Results of P70 (b) Results of P89

Figure 7 Best results of P70 and P89 by SA algorithms with ten runs

The following section focuses on the comparison campaign among the tested four algorithms. Table 4 shows the average results of hyper volume ratio and Unary Epsilon Indicator under four termination criteria. Each cell in this table contains the average

results of four cases and 10 runs. In other words, each cell reports the average result of 40 cases. From the table, it is observed that, from the shortest elapsed CPU time of $\tau = 10$ the proposed MOCC is the best performer regarding *HVR*, and I_ε^1. If on ranks algorithms using the *HVR* indicator; MOCC and RSA are the best one and second best ones whereas NSGA-II and ABC are the worst and the second worst ones for each termination criterion. Specifically, the overall *HVR* of MOCC is 0.74, 0.79, 0.83 and 0.85 that is significantly larger than the three comparison algorithms. Perhaps more interestingly MOCC achieves all the best average results for all the five large-sized problems regarding the *HVR* indicator and I_ε^1indicator. All these results indicate the superiority of the proposed MOCC regarding the *HVR* and I_ε^1 indicators.

Table 4 Computational results for RALB problem

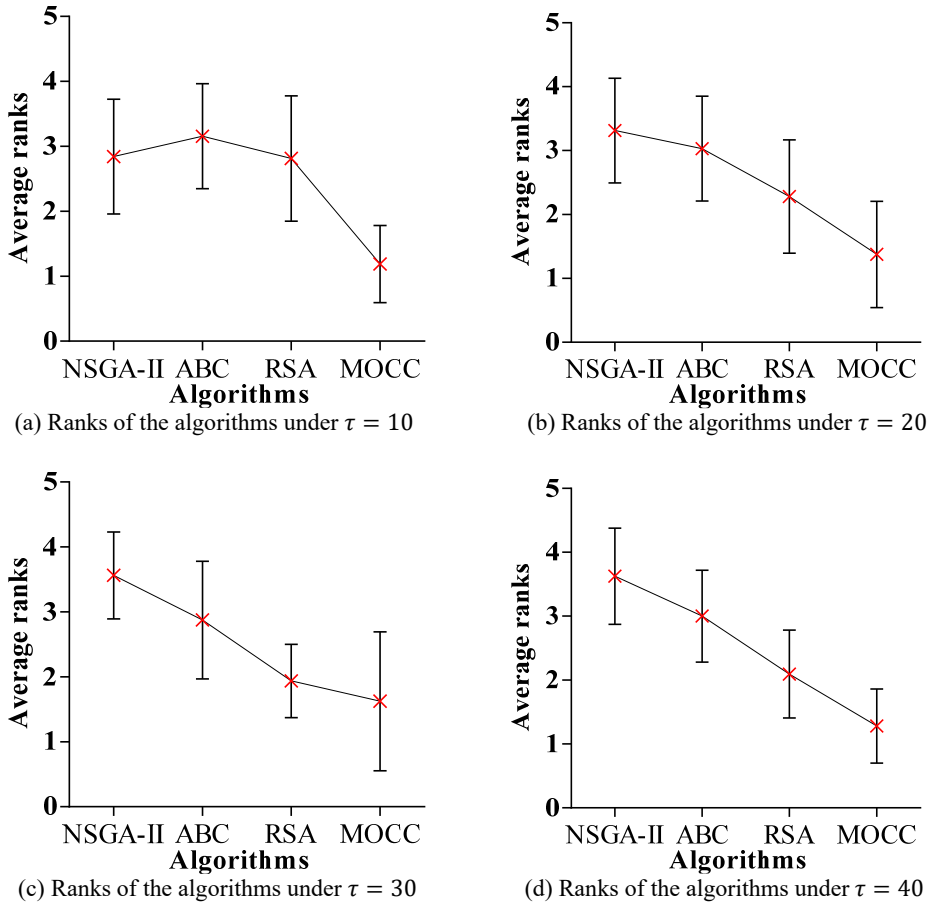
Problem	HVR				I_ε^1			
	NSGA-II	ABC	SA	MOCC	NSGA-II	ABC	SA	MOCC
$\tau = 10$								
P25	0.92	0.89	0.95	0.96	1.08	1.09	1.06	1.05
P35	0.78	0.74	0.83	0.84	1.26	1.27	1.24	1.17
P53	0.78	0.75	0.79	0.82	1.24	1.23	1.26	1.21
P70	0.65	0.65	0.73	0.79	1.58	1.53	1.52	1.36
P89	0.60	0.61	0.67	0.74	1.47	1.42	1.43	1.30
P111	0.54	0.56	0.53	0.68	1.68	1.60	1.75	1.46
P148	0.47	0.50	0.43	0.62	1.70	1.63	1.81	1.55
P297	0.29	0.30	0.13	0.48	2.02	1.93	2.41	1.86
Avg.	0.63	0.62	0.63	**0.74**	1.50	1.46	1.56	**1.37**
$\tau = 20$								
P25	0.92	0.92	0.96	0.96	1.08	1.08	1.05	1.05
P35	0.83	0.79	0.85	0.84	1.20	1.22	1.21	1.15
P53	0.78	0.78	0.82	0.83	1.25	1.21	1.23	1.19
P70	0.66	0.70	0.76	0.83	1.54	1.43	1.42	1.26
P89	0.62	0.67	0.72	0.77	1.43	1.33	1.36	1.26
P111	0.59	0.65	0.71	0.77	1.58	1.47	1.48	1.27
P148	0.51	0.58	0.63	0.72	1.60	1.49	1.52	1.32
P297	0.37	0.39	0.41	0.60	1.85	1.76	1.84	1.63
Avg.	0.66	0.68	0.73	**0.79**	1.44	1.37	1.39	**1.27**
$\tau = 30$								
P25	0.94	0.93	0.96	0.97	1.07	1.07	1.04	1.04
P35	0.80	0.82	0.87	0.85	1.22	1.19	1.20	1.16
P53	0.75	0.81	0.83	0.84	1.28	1.19	1.21	1.18
P70	0.71	0.74	0.81	0.84	1.44	1.34	1.34	1.24
P89	0.66	0.70	0.78	0.81	1.39	1.32	1.30	1.21
P111	0.59	0.65	0.73	0.79	1.53	1.42	1.41	1.22
P148	0.54	0.63	0.71	0.77	1.52	1.41	1.40	1.22
P297	0.41	0.48	0.61	0.74	1.64	1.54	1.46	1.31
Avg.	0.68	0.72	0.79	**0.83**	1.39	1.31	1.29	**1.20**
$\tau = 40$								
P25	0.94	0.94	0.96	0.97	1.07	1.07	1.04	1.03
P35	0.81	0.81	0.87	0.86	1.21	1.20	1.18	1.15
P53	0.78	0.83	0.85	0.87	1.25	1.18	1.21	1.15
P70	0.69	0.70	0.82	0.83	1.43	1.39	1.33	1.23
P89	0.64	0.73	0.79	0.84	1.41	1.29	1.28	1.20
P111	0.63	0.74	0.79	0.84	1.50	1.32	1.38	1.17
P148	0.60	0.70	0.80	0.81	1.47	1.36	1.32	1.19
P297	0.39	0.52	0.64	0.75	1.65	1.49	1.43	1.26
Avg.	0.69	0.75	0.81	**0.85**	1.37	1.29	1.27	**1.17**

*Best average results in bold.

Though the results in the previous table show quite a clear performance difference, it is still prudent to carry out statistical analysis to ascertain whether the observed

differences are statistically significant. This research uses non-parametric Friedman rank-based analysis (Friedman, 1937) since the normality of the residuals is violated, which is observed by an initial analysis with parametric ANOVA technique. This situation is caused by the big differences of algorithms' performance on the different tested problems. In Friedman rank-based analysis, the results are replaced with ranks. In this paper, the best result is given a rank of 1 and the worst result is provided with a rank of 4. It is worth noting that Friedman rank-based analysis neglects the real differences in the indicators, and the difference might be smaller or greater with respect to the equivalent ANOVA test. However, given the non-normal nature of the results it still appears as the most suitable method for comparison. There is still another underlying problem that some indicators prefer lower values and others prefer higher values. To obtain a common format, the values of the indicators that prefer higher values are modified utilizing the reciprocals of these values. After this transformation, the algorithm that ranks one is the best performer regardless of indicator. This research proposes two quantitative performance indicators and four termination criteria, resulting in twenty statistical analysis results. Instead of detailing the statistical results, it is chosen to primarily present the average ranks of the algorithms for different indicators and termination criteria.

The primary concern is to compare the performances of algorithms under four elapsed CPU time and the *HVR* indicator is selected as response variable for being the most appropriate scalar indicator as stated by Zitzler et al. (2001). Figure 8 (a-d) depicts the average ranks of the algorithms along with 95% minimal significant difference confidence intervals under $\tau = 10$ (a), $\tau = 20$ (b), $\tau = 30$ (c) and $\tau = 40$ (d). It is observed that MOCC ranks first under the four termination criteria indicating that MOCC is the best performer. The remaining algorithms, SA, ABC and NSGA-II rank second, third and last.



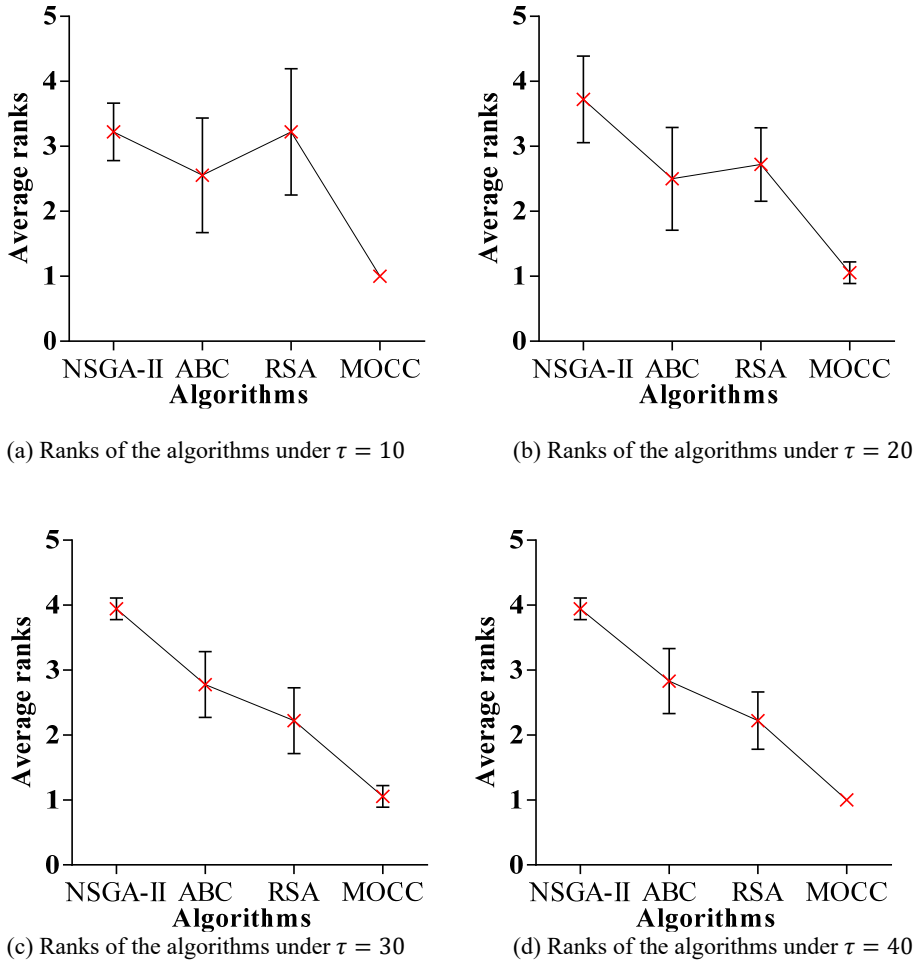
(a) Ranks of the algorithms under $\tau = 10$ (b) Ranks of the algorithms under $\tau = 20$

(c) Ranks of the algorithms under $\tau = 30$ (d) Ranks of the algorithms under $\tau = 40$

Figure 8 Means plot of the average ranks and 95% confidence intervals of algorithms regarding *HVR* indicator

Figure 9 (a-b) also depicts the average ranks of the algorithms along with 95% minimal significant difference confidence intervals for I_ε^1. It is also observed that the proposed MOCC is the best performer under the four termination criteria utilizing this indicator. ABC algorithm shows a clear superiority over NSGA-II and RSA becomes the second best performer when $\tau = 30$ and $\tau = 40$. These statistical tests further validate and strengthen the results in Table 4. To further show the spatial performance, Figure 10 (a-c) also exhibit the Diff-EAF between MOCC and NSGA-II, MOCC and ABC, and MOCC and RSA for instance P89 with 16 workstations under $Nt \times Nt \times 20$ millisecond termination criterion. This paper employs 50 replicates for each algorithm to obtain the three figures above. These figures directly provide information about the spatial performance of the compared algorithms. It is observed that MOCC outperforms the NGGA-II and ABC for almost all the objective space and the advantage of MOCC over NGGA-II and ABC is quite clear. Based on Figure 10c, the results are ambiguous. It is clear that MOCC achieves better results at extremes of the frontier whereas RSA performs better in the central part of the objective space. In that case, RSA performs better in fewer regions, and it is reasonable to state that MOCC outperforms RSA. Using this graphic tool, MOCC is confirmed to outperform NSGA-II and ABC by a significant

margin and at least be able to obtain competing results compared with RSA.



(a) Ranks of the algorithms under $\tau = 10$

(b) Ranks of the algorithms under $\tau = 20$

(c) Ranks of the algorithms under $\tau = 30$

(d) Ranks of the algorithms under $\tau = 40$

Figure 9 Means plot of the average ranks and 95% confidence intervals of algorithms regarding I_ε^1 indicators

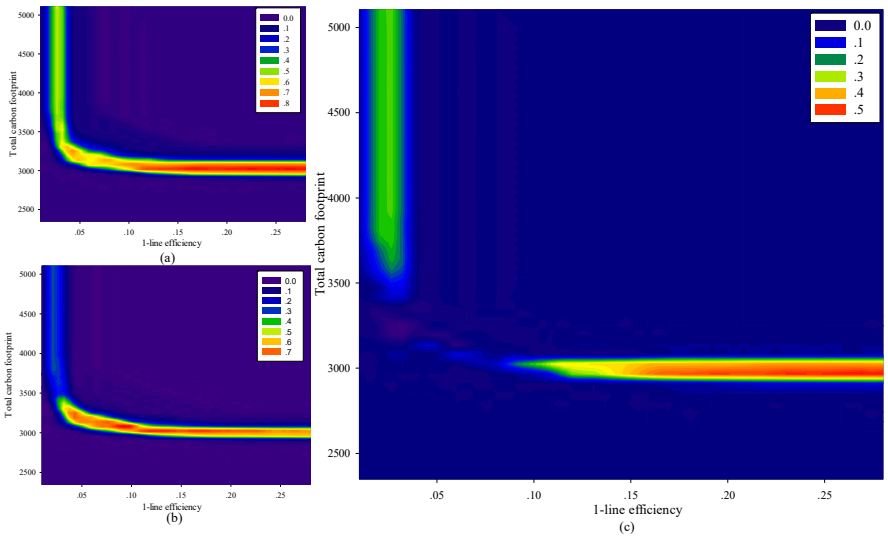


Figure 10 Differences between Empirical Attainment Functions for MOCC vs NSGA-II (a), MOCC vs ABC (b) and MOCC vs RSA (c).

6. Conclusion and future research

Robotic assembly lines are becoming increasingly popular for the assembly of different types of products and these types of assembly lines are extensively used in industry. Recently, the reduction carbon footprint as a new overall environment criterion has become critical for reducing environmental impacts. This paper provides a method to simultaneously tackle the carbon footprint reduction in terms of energy consumption and line efficiency maximization in robotic assembly line systems.

This paper has two main contributions. Firstly, a multi-objective model is developed to minimize carbon footprint reduction and maximize line efficiency. This model considers the carbon footprint caused by the total energy consumption of the line. Two types of energy consumption modes are considered: operation and standby mode. Secondly, a multi-objective co-operative co-evolutionary algorithm (MOCC) is developed and improved to solve this problem. The MOCC contains two sub-swarms to optimize the task assignment and robot allocation. The extending operator is modified to increase local search capacity and avoid the algorithm being trapped in a local optimum. A comprehensive study on a set of 32 benchmark problems is carried out and the proposed MOCC is compared with existing well-known algorithms. Two quantitative indicators and one graphical indicator are proposed to measure the obtained Pareto sets. Computational and statistical results using two Pareto compliant indicators and one graphic indicator demonstrate the superiority of the proposed MOCC.

Energy consumption cost is one of the major expenses for robotic assembly lines and reducing the energy usage will assist companies to remain competitive in the market. The results obtained from this study can be used to reduce the cost of energy consumed and will assist companies in focusing on the correct areas of improvement. In the proposed model, robots can be allocated to any station and a task can be operated by any robot, which is not necessarily true. This research differs from current state, since the majority of the research published to-date focuses on minimizing cycle time and number of workstations and no work focuses on simultaneously optimizing carbon foot print and line efficiency of a robotic assembly lines. This model will have significant managerial implications. It will assist managers in designing/redesigning efficient robotic assembly lines with respect to carbon footprint minimization and line efficiency maximization.

This research mainly considers the carbon footprint resulting from power consumption and other kinds of carbon footprint caused by material consumption can be considered in the future. This research can be further extended by taking the practical constraints into account, including the allocation of robots and assignment of tasks. A realistic example from robotic assembly lines focusing on the carbon footprint is also interesting to narrow the gap between research and real applications. The algorithm can be further improved by including energy consumption during maintenance operations and effects of failure of the resources in the system. Lastly, the study reported on MOCC is minimal and it shows promising results for optimizing two sub-problems simultaneously and the further exploration of this method and the applications of the algorithm to other research areas may be good research avenues.

Acknowledgment

This research work is funded by the National Natural Science Foundation of China (Grant No. 51275366) (Qiuhua Tang).

References

Aghajani, M., Ghodsi, R., Javadi, B., 2014. Balancing of robotic mixed-model two-sided assembly line with robot setup times. The International Journal of Advanced Manufacturing Technology, 74, 1005-1016.

Bertoldi, P., Aebischer, B., Edlington, C., Hershberg, C., Lebot, B., Lin, J., Marker, T., Meier, A., Nakagami, H., Shibata, Y., 2002. Standby power use: How big is the problem? What policies and technical solutions can address it? in: Proc. of the Summer Conference of the American Council for an Energy-Efficient Economy (ACEEE),Washington,DC.

Ciavotta, M., Minella, G., Ruiz, R., 2013. Multi-objective sequence dependent setup times permutation flowshop: A new algorithm and a comprehensive study. European Journal of Operational Research, 227, 301-313.

da Fonseca, V.G., Fonseca, C.M., Hall, A.O., 2001. Inferential performance assessment of stochastic optimisers and the attainment function, In: Proceedings of International Conference on Evolutionary Multi-Criterion Optimization. Springer, pp. 213-225.

Dai, M., Tang, D., Giret, A., Salido, M.A., Li, W.D., 2013. Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. Robotics and Computer-Integrated Manufacturing, 29, 418-429.

Dang, Q.-V., Nielsen, I., Steger-Jensen, K., 2012. Multi-objective Genetic Algorithm for Real-World Mobile Robot Scheduling Problem, In: Proceedings of IFIP International Conference on Advances in Production Management Systems. Springer, pp. 518-525.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6, 182-197.

Friedman, M., 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association, 32, 675-701.

Fysikopoulos, A., Anagnostakis, D., Salonitis, K., Chryssolouris, G., 2012. An empirical study of the energy consumption in automotive assembly. Procedia CIRP, 3, 477-482.

Gao, J., Sun, L., Wang, L., Gen, M., 2009. An efficient approach for type II robotic assembly line balancing problems. Computers & Industrial Engineering, 56, 1065-1080.

Lee, T.O., Kim, Y., Kim, Y.K., 2001. Two-sided assembly line balancing to maximize work relatedness and slackness. Computers & Industrial Engineering, 40, 273-292.

Levitin, G., Rubinovitz, J., Shnits, B., 2006. A genetic algorithm for robotic assembly line balancing. European Journal of Operational Research, 168, 811-825.

Li, C., Tang, Y., Cui, L., Li, P., 2015. A quantitative approach to analyze carbon emissions of CNC-based machining systems. Journal of Intelligent Manufacturing, 26, 911-922.

Li, Z., Janardhanan, M.N., Tang, Q., Nielsen, P., 2016a. Co-evolutionary particle swarm optimization algorithm for two-sided robotic assembly line balancing problem. Advances in Mechanical Engineering, 8, doi:10.1177/1687814016667907.

Li, Z., Tang, Q., Zhang, L., 2016b. Minimizing energy consumption and cycle time in two-sided robotic assembly line systems using restarted simulated annealing algorithm. Journal of Cleaner Production, 135,

508-522.
Li, Z., Tang, Q., Zhang, L., 2016c. Minimizing the cycle time in two-sided assembly lines with assignment restrictions: improvements and a simple algorithm. Mathematical Problems in Engineering, Article ID 4536426.
Lin, W., Yu, D., Zhang, C., Liu, X., Zhang, S., Tian, Y., Liu, S., Xie, Z., 2015. A multi-objective teaching–learning-based optimization algorithm to scheduling in turning processes for minimizing makespan and carbon footprint. Journal of Cleaner Production, 101, 337-347.
Liu, C.-H., 2014. Approximate trade-off between minimisation of total weighted tardiness and minimisation of carbon dioxide (CO2) emissions in bi-criteria batch scheduling problem. International Journal of Computer Integrated Manufacturing, 27, 759-771.
López-Ibánez, M., Paquete, L., Stützle, T., 2006. Hybrid population-based algorithms for the bi-objective quadratic assignment problem. Journal of Mathematical Modelling and Algorithms, 5, 111-137.
Montgomery, D.C., 2008. Design and analysis of experiments. John Wiley & Sons, New York, USA.
Nilakantan, J.M., Huang, G.Q., Ponnambalam, S., 2015a. An investigation on minimizing cycle time and total energy consumption in robotic assembly line systems. Journal of Cleaner Production, 90, 311-325.
Nilakantan, J.M., Nielsen, I., Ponnambalam, S.G., Venkataramanaiah, S., 2017. Differential evolution algorithm for solving RALB problem using cost- and time-based models. The International Journal of Advanced Manufacturing Technology, 89, 311-332.
Nilakantan, J.M., Ponnambalam, S., 2016. Robotic U-shaped assembly line balancing using particle swarm optimization. Engineering Optimization, 48, 231-252.
Nilakantan, J.M., Ponnambalam, S., Huang, G.Q., 2015b. Minimizing energy consumption in a U-shaped robotic assembly line, In: Proceedings of International Conference on Advanced Mechatronic Systems (ICAMechS). IEEE, pp. 119-124.
Nilakantan, J.M., Ponnambalam, S., Jawahar, N., Kanagaraj, G., 2015c. Bio-inspired search algorithms to solve robotic assembly line balancing problems. Neural Computing and Applications, 26, 1379-1393.
Nilakantan, M.J., Ponnambalam, S., Jawahar, N., 2016. Design of energy efficient RAL system using evolutionary algorithms. Engineering Computations, 33, 580-602.
Rubinovitz, J., Bukchin, J., 1991. Design and balancing of robotic assembly lines. Society of Manufacturing Engineers.
Rubinovitz, J., Bukchin, J., Lenz, E., 1993. RALB–A heuristic algorithm for design and balancing of robotic assembly lines. CIRP Annals-Manufacturing Technology, 42, 497-500.
Saif, U., Guan, Z., Liu, W., Wang, B., Zhang, C., 2014. Multi-objective artificial bee colony algorithm for simultaneous sequencing and balancing of mixed model assembly line. The International Journal of Advanced Manufacturing Technology, 75, 1809-1827.
Tang, D., Dai, M., 2015. Energy-efficient approach to minimizing the energy consumption in an extended job-shop scheduling problem. Chinese Journal of Mechanical Engineering, 28, 1048-1055.
Tang, Q., Li, Z., Zhang, L., 2016. An effective discrete artificial bee colony algorithm with idle time reduction techniques for two-sided assembly line balancing problem of type-II. Computers & Industrial Engineering, 97, 146-156.
Yi, Q., Li, C., Tang, Y., Wang, Q., 2012. A new operational framework to job shop scheduling for reducing carbon emissions, In: Proceedings of IEEE International Conference on Automation Science and Engineering (CASE).
Yoosefelahi, A., Aminnayeri, M., Mosadegh, H., Ardakani, H.D., 2012. Type II robotic assembly line balancing problem: An evolution strategies algorithm for a multi-objective model. Journal of

Manufacturing Systems, 31, 139-151.
Zhao, W., Alam, S., Abbass, H.A., 2014. MOCCA-II: A multi-objective co-operative co-evolutionary algorithm. Applied Soft Computing, 23, 407-416.
Zitzler, E., Laumanns, M., Thiele, L., 2001. SPEA2: Improving the strength Pareto evolutionary algorithm, Swiss Federal Institute Techonology: Zurich, Switzerland, pp. 95-100.