Zhang, Z.; Tang, Q.; Ruiz García, R.; Zhang, L. (2020). Ergonomic risk and cycle time minimization for the U-shaped worker assignment assembly line balancing problem: A multi-objective approach. Computers & Operations Research. 118:1-15. https://doi.org/10.1016/j.cor.2020.104905

Ergonomic risk and cycle time minimization for the U-shaped worker assignment assembly line balancing problem: A multi-objective approach

Zikai Zhang[1, 2], QiuHua Tang[1, 2], Rubén Ruiz[3], Liping Zhang[1, 2]

1: Key Laboratory of Metallurgical Equipment and Control Technology, Wuhan University of Science and Technology, China.

2: Hubei Key Laboratory of Mechanical Transmission and Manufacturing Engineering, Wuhan University of Science and Technology.

3: Grupo de Sistemas de Optimización Aplicada, Instituto Tecnológico de Informática, Ciudad Politécnica de la Innovación, Edificio 8G, Acc. B. Universitat Politècnica de València, Camino de Vera s/n, 46022, València, Spain.

Abstract: Workers still perform the bulk of operations in the manufacturing industry. The consideration of the assignment of workers and the reduction of ergonomic risks in U-shaped assembly lines is of paramount importance. However, the objectives of efficient task and worker assignment and a reduction in ergonomic risks are not usually correlated. Moreover, there is limited research in the existing literature into multi-objective approaches in U-shaped assembly lines. We formulate a U-shaped assembly worker assignment and balancing problem to simultaneously minimize cycle times and ergonomic risks. In addition, and due to its simplicity and successful results in flow shop scheduling problems, a Restarted Iterated Pareto Greedy algorithm is designed to optimize both objectives. In this algorithm, a problem-specific heuristic-based initialization is extended to improve the initial solution. Two precedence-based greedy and local search phases are developed to exploit the space around the current solution. Finally, a restart mechanism is proposed to help the algorithm escape from local optima. Comprehensive computational results, supported by detailed statistical analyses, suggest that the proposed multi-objective algorithm outperforms existing methods on a large number of benchmark instances.

Keywords: U-shaped assembly line, worker assignment, ergonomic risks, multi-objectives.

1. Introduction

In the early 20th century, engineers at Ford Motor Company managed to divide the complex assembly process into many simple tasks to reduce the technical requirements for workers. This improvement quickly cut the completion time of a car from 718 hours to a mere 93 minutes (Yorke, 2017). Since then, the assembly line has become the primary production model in manufacturing industries, such as automobiles, food, toys and furniture. An assembly line consists of m workstations in which workers perform some specific tasks, and the corresponding problem, named the assembly line balancing problem (ALBP), focuses on the allocation of a set of n tasks to these workstations to achieve one or several given objectives. These tasks are characterized by the deterministic processing times and precedence relationships among them (Bukchin and Raviv, 2018). The allocation of these tasks must not only meet the precedence relationships, but also satisfy a limitation of the given cycle time on each workstation.

To further improve productivity and quality, some enterprises have begun to improve on the layout of assembly lines. Compared with the simple straight assembly line, the U-shaped assembly line contains not only regular workstations to execute tasks on either the entrance or the exit subline, but also cross workstations to perform tasks on both entrance and exit sublines. Particularly, a task may be assigned as long as its immediate predecessors/successors have been assigned, and as a result, this added flexibility offers a significantly higher level of productivity (Baykasoğlu and Özbakır, 2006). As shown in Fig. 1(a), nine tasks with precedence constraints need to be allocated to three workstations. A typical layout of a U-shaped line is shown in Fig. 1(b). This U-shaped line has nine tasks and three workstations where stations 1 and 2 are cross workstations. Specifically,

tasks 1, 2, 3, 5, 4 and 8 are allocated to the entrance of these stations while the other tasks are allocated to the exit. Taking workstation 1 for example, the worker performs task 1 in the entrance and then goes to the exit to process tasks 7 and 9. It is worth noting that according to the objective function, the corresponding U-shaped assembly line balancing problem (UALBP), first formulated by Miltenburg and Wijingaard (1994), can fall into a further four (Rabbani et al., 2012). Type-I deals with the minimization of the number of workstations for a given cycle time. Type-II aims to optimize the cycle time for a given number of workstations. Type-E maximizes line efficiency when both the cycle time and number of workstations are unknown. Type-F seeks a feasible balance plan when the cycle time as well as the number of workstations are both given. In this paper, we study the type-II category with the intention of optimizing the cycle time when the number of workstations is known.

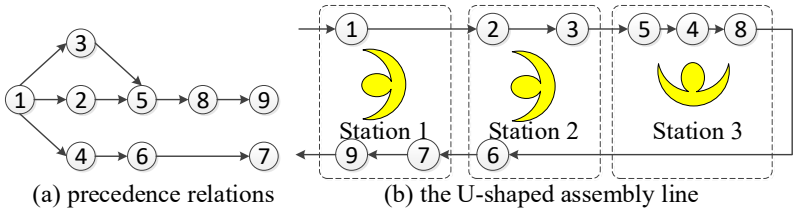


Fig. 1. Example layout of a U-shaped assembly line.

In real production settings, assembly operations are flexible and need to be organized so that workers' hearing and sight are not impeded and they have the ability to sense minute vibrations. Therefore, most current U-shaped lines are manual due to the flexibility of human workers. However, most of the published work assumes that workers have the same abilities and skills, and hence the processing time of each task is fixed. This is not realistic. Not all workers can process a given task at the same speed and some workers might not be able to perform some tasks at all due to a need for qualifications and/or technical or physical limitations. Therefore, when balancing a U-shaped line, not only should tasks be allocated to workstations, but also the assignment of workers needs to be considered. This problem is referred to as the U-shaped assembly line worker assignment and balancing problem (UALWABP, Oksuz et al. (2017)).

Moreover, unchecked ergonomic risks in unfavorable working conditions will lead to work-related musculoskeletal disorders (WMSDs) in workers. Generally, poor working conditions involve workload factors and environmental factors (Otto and Battaia, 2017). Specifically, the workload factors include the lifting of heavy loads, awkward postures, sitting or standing for long periods of time, repetitive movements and/or vibrations. Environmental factors refer to temperature, humidity, noise and lighting. According to some survey reports, in Europe about 44 million workers suffer from occupational musculoskeletal disorders (Otto and Battaia, 2017); In the US, approximately 315,000 cases of WMSDs required a median of 10 days away from work in 2008 (Otto and Scholl, 2011). Thus, the reduction of ergonomic risks has become an important topic in recent years and is considered as the second objective to optimize in this paper. It has to be mentioned that, to the best of our knowledge, the UALWABP Type-II with cycle time and ergonomic risk minimization problem has not been dealt with before in the scientific literature.

The remainder of this paper is organized as follows: A review of the related literature is given in Section 2. This new problem involving ergonomic risks is formulated in Section 3. Then, the proposed Restarted Iterated Pareto Greedy algorithm to simultaneously minimize cycle time and ergonomic risk is described in Section 4. Experimental results are reported in Section 5. Finally, conclusions and future work are discussed in Section 6.

2. Literature review

2.1. Current state of the art for the U-shaped assembly line

The UALBP is a classic problem which was first formulated by Miltenburg and Wijingaard (1994). The authors built a mathematical model to describe this problem and then developed a dynamic programming procedure to determine the optimal balancing for small-sized instances. For larger cases, they proposed a greedy heuristic based on a ranked positional weight technique. Due to the high flexibility of U-lines, there has been a growing interest in this area from the scientific community. In recent years, Avikal et al. (2013) proposed a critical path method, assigning tasks to workstations, to improve labor productivity on the U-line. Fattahi et al. (2013) presented a formulation to minimize the number of workstations in U-lines. This approach employed three types of logic cuts to exploit the inherent structure of the problem. Nourmohammadi et al. (2013) developed an imperialist competitive algorithm (ICA) to address the multi-objective UALBP. They aimed to optimize line efficiency and workload variation. Alavidoost et al. (2015) utilized a fuzzy adaptive genetic algorithm where a one fifth success rule was deployed to improve the performance of the algorithm. Hazır and Dolgui (2015) considered the unfixed processing time of tasks and employed a decomposition based solution algorithm for the UALBP. Alavidoost et al. (2016) employed triangular fuzzy numbers to represent the uncertainty of processing times and proposed a two-phase interactive fuzzy programming approach to minimize the number of workstations and fuzzy cycle time. Aydoğan et al. (2016) designed a particle swarm optimization (PSO) algorithm to balance stochastic U-lines. Li, M. et al. (2017) proposed a rule-based heuristic approach, which systematically considered task selection, task assignment and task exchange rules together to minimize the cycle time of the UALBP. Sahin and Kellegoz (2017) designed a grouped algorithm to maximise the production rate of the UALBP. This algorithm merged a genetic algorithm (GA) with a simulated annealing method (SA) to improve performance.

All the aforementioned papers balance U-shaped assembly lines by allocating tasks to workstations, but ignore the drastic effect of different workers' skills on processing times. To our knowledge, there is only two piece of research by Oksuz et al. (2017) and Zhang et al. (2018) studying the UALWABP. The former formulated a nonlinear model of type-E UALWABP which was later linearized. Furthermore, two meta-heuristics, the artificial bee colony algorithm (ABC) and genetic algorithm, are proposed to maximize line efficiency. The later proposed an enhanced migrating birds optimization algorithm to minimize the cycle time of UALWABP. Other papers study worker assignment in the ALBP, named the assembly line worker assignment and balancing problem (ALWABP). Chaves et al. (2007) firstly introduced and then formulated a mathematical model. Later, Chaves et al. (2009) designed a hybrid clustering search method to solve this problem. This algorithm employed an iterated local search to generate solutions and explore their neighborhoods, and then employed some clusters to store these solutions. Blum and Miralles (2011) proposed beam search (BS) to minimize the cycle time of the ALWABP. Mutlu et al. (2013) developed an iterative genetic algorithm (IGA) for the ALWABP. Borba and Ritt (2014) presented a new MIP model and designed a heuristic and a branch-and-bound algorithm for the type-II ALWABP. Vila and Pereira (2014) also proposed a branch-and-bound algorithm for the type-I and II ALWABP. Ramezanian and Ezzatpanah (2015) presented a goal programming approach and ICA to minimize the cycle time and operating cost of the mixed-model ALWABP. Sungur and Yavuz

(2015) formulated an integer linear model to optimize the total cost of this problem where they assumed that more highly qualified workers might substitute lower qualified ones with a higher cost. Zacharia and Nearchou (2016) proposed a multi-objective evolutionary algorithm to optimize the cycle time and smoothness of ALWABP. Pereira (2018) assumed that the processing time is between a lower and an upper bound and then investigated the relationship with other interval data minmax regret problem.

2.2. Ergonomic risks in straight/U-shaped assembly line

In recent years, some studies have considered the ergonomic risks to workers in the assembly line balancing problem in order to reduce the risks of musculoskeletal diseases. To determine the ergonomic risks, some risk assessment methods for working conditions have been already proposed. According to the working conditions, these methods can be categorized into four types: (1) methods for lifting of heavy loads including the Job Strain Index (JSI-L) (Liles et al., 1984) and the National Institute for Occupational Safety and Health lifting equation (NIOSH-Eq) (Waters et al., 1993); (2) methods for awkward postures involving the Ovako Working Analysis System (OWAS) (Karhu et al., 1977), the Rapid Upper Limb Assessment (RULA) (McAtamney and Nigel Corlett, 1993) and the Rapid Entire Body Assessment (REBA) (Hignett and McAtamney, 2000); (3) methods for repetitive work are addressed by the Occupational Repetitive Action tool (OCRA) (Occhipinti, 1998) and European Assembly WorkSheet (EAWS) (Schaub et al., 2013); (4) methods for noise assessment include the Daily Noise Dosage (DND) (Aryanezhad et al., 2009). Among them, the OCRA method deals with ergonomic risks by calculating the fatigue of repetitive actions. This method comprehensively evaluates the ergonomics of posture, force, repetitiveness and additional risks. Specifically in U-shaped assembly lines, since workers perform most tasks with their upper limbs, the OCRA method is often used for modeling ergonomic risks. The relative literature about OCRA in assembly line is briefly reviewed in the following.

Otto and Scholl (2011) introduced some methods to estimate the ergonomic risks of each workstation in assembly line balancing. Specifically, they used a revised NIOSH equation to estimate the risk of manual handling, OCRA to upper limbs and EAWS to the whole body. They proposed a two-stage heuristic procedure to find a solution with a minimal number of workstations with high ergonomic risks. Battini et al. (2011) analyzed how to relate the ergonomics with assembly lines and developed a theoretical framework to comprehensive consider the technological variables related to assembly lines, environmental variables and ergonomic evaluation. Battini et al. (2016) developed a model to incorporate ergonomics into the integrated assembly line balancing and parts feeding problem. In their model, they reserved enough rest time for operators as a function of energy expenditure to reduce the ergonomic risks. Bautista et al. (2016) formulated several models to solve the ALBP considering temporal, spatial and ergonomic attributes. Akyol and Baykasoğlu (2016) employed the OCRA method to calculate the ergonomic risk of workstations and then utilized a multiple-rule based constructive randomized search algorithm to find a solution with the lowest ergonomic risks for the ALWABP. Bortolini et al. (2017) proposed a multi-objective model for the ALBP considering component picking and ergonomic risks. This model assigned tasks to workstations and determined the storage location of each component between the locations at the different workstations to minimize the assembly line takt time and ergonomic risk. Botti et al. (2017) designed an effective and efficient assembly line balancing plan that meets ergonomic requirements and lean principles. Otto and Battaia (2017) summarized the existing optimization methods for the UALBP and proposed a job rotation scheduling method considering ergonomic risks. Tiacci and

Mimmi (2018) integrated the ergonomic risks evaluation into mixed-model stochastic assembly line balancing and sequencing, and employed genetic algorithm to reduce the cycle time and ergonomic load. Shin and Park (2019) proposed a well-balanced schedule with the lowest ergonomic risks optimizing the total physical stress on workers. Finco et al. (2019) considered human energy expenditure as ergonomic aspects and integrated it into assembly lines, and designed a heuristic approach to minimize the smoothness index.

Based on this review, and to the best of our knowledge, there appears to be no prior research focusing on the UALWABP with the simultaneous consideration of ergonomic risks and cycle time minimization. Current research only optimizes either the cycle time or ergonomic risk while practical production problems often need a joint consideration of both objectives. Managers wish to reduce cycle times to improve production efficiency, but workers are more concerned with their health and having low ergonomic risks. Cycle times and ergonomic risk minimization are often conflicting goals. That is, a minimum cycle time might lead to high ergonomic risks and vice-versa. Thus, it is important to consider these two objectives simultaneously. There are three main methods for tackling multi-objective problems: the weighted approach, lexicographical approach and Pareto front. With regard to the properties of these methods, the weighted approach is greatly influenced by the magnitude of the objective functions, and the weight value itself is determined artificially or optimized through computational experiments. The lexicographical approach first optimizes the upper-level objective and then the lower-level, resulting in the first level interfering in the optimization of the following objective. Different from the above two methods, the Pareto front can achieve a trade-off between the objectives to obtain non-dominated solutions. Thus, the Pareto front is selected to deal with this bi-objective problem. Furthermore, since the iterated greedy algorithm has been successfully applied to single and multi-objective flow shops (Minella et al., 2011), a more generalized version of the assembly line, it is here extended to tackle the UALWABP considering cycle time and ergonomic risk minimization.

Therefore, this paper principally contains the following contributions.

- A new UALWABP with the objectives of ergonomic risk and cycle time is formulated.
- A Restarted Iterated Pareto Greedy algorithm (RIPG, (Minella et al., 2011)) is extended to tackle this problem. This algorithm includes heuristic rules-based initialization and precedence-based greedy and local search phases to obtain many trade-off solutions in a small amount of time.

3. Problem formulation and mathematical model

This section formulates the worker assignment and balancing problem within the type-II U-shaped assembly line to minimize cycle time and ergonomic risks. A mathematical model is presented. Recall that in type-II, both worker and task assignments need to be solved. It must be pointed out that the processing time of a given task is fully dependent on the workers since different workers have different skills and abilities. The assumptions for the type-II UALWABP are stated as follows:

1) Walking times of workers within workstations are negligible and therefore are ignored.
2) Workers can perform all tasks with potentially different processing times.
3) Workers can process tasks from the entrance subline to the exit in a crossover workstation.
4) Only one type of product is assembled in this line.
5) The precedence relations among tasks are known and the allocation of tasks needs to satisfy precedence relations.

6) Each task is executed with a type of posture (position and movement of the upper limbs of a worker).

3.1. The notation and input data of the studied UALWABP go as follows:

NI	Number of tasks.
NW	Number of workstations.
NH	Number of workers.
I	Set of all tasks, $\|I\| = NI$.
J	Set of all workstations, $\|J\| = NW$.
H	Set of all workers, $\|H\| = NH$.
i, p, s	Indices referring to tasks, $i, p, s = 1, 2, 3,\ldots, NI$.
j	Workstation index , $j = 1, 2, 3,\ldots, NW$.
h	Worker index, $h = 1, 2, 3,\ldots, NH$.
W	Set of precedence relationships. $(p, s) \in W$, where p is an immediate predecessor task of task s and task s is an immediate successor of task p.
t_{ih}	Processing time of task i when performed by worker h.
M	A large positive number.

3.2. Decision variables

The proposed mathematical model entails the following decision variables:

X_{ijh}	Binary variable that takes value 1 if task i is assigned to workstation j at the entrance subline and is carried out by worker h and 0 otherwise.
Y_{ijh}	Binary variable that takes value 1 if task i is assigned to workstation j at the exit subline and is carried out by worker h and 0 otherwise.
Z_{hj}	Binary variable that takes value 1 if worker h is assigned to workstation j and 0 otherwise.
CT	Cycle time.
ER_j	Ergonomic risk value of workstation j.
TER	Total ergonomic risk.

3.3. Mathematical model

Objective functions:

$$\min CT \quad (1)$$

$$\min TER = \min \sum_{j}^{NW} ER_j \quad (2)$$

Subject to:

$$\sum_{i=1}^{NI}\sum_{h=1}^{NH}\left[t_{ih} \times (X_{ijh} + Y_{ijh})\right] \leq CT, \forall j \in J \quad (3)$$

$$\sum_{j=1}^{NW}\sum_{h=1}^{NH}(X_{ijh} + Y_{ijh}) = 1, \forall i \in I \quad (4)$$

$$\sum_{i=1}^{NI}\sum_{h=1}^{NH}(X_{ijh} + Y_{ijh}) \geq 1, \forall j \in J \quad (5)$$

$$\sum_{j=1}^{NW}\sum_{h=1}^{NH}(M - j + 1) \times \left(X_{pjh} - X_{sjh}\right) \geq 0, \forall\, (p, s) \in W \quad (6)$$

$$\sum_{j=1}^{NW}\sum_{h=1}^{NH}(M - j + 1) \times \left(Y_{sjh} - Y_{pjh}\right) \geq 0, \forall\, (p, s) \in W \quad (7)$$

$$X_{ijh} + Y_{ijh} \leq Z_{hj}, \forall i \in I, j \in J, h \in H \quad (8)$$

$$\sum_{h=1}^{NH} Z_{hj} = 1, \forall j \in J \quad (9)$$

$$\sum_{j=1}^{NW} Z_{hj} = 1, \forall h \in H \quad (10)$$

$$X_{ijh} = \{0, 1\}, \forall i \in I, j \in J, h \in H \quad (11)$$

$$Y_{ijh} = \{0, 1\}, \forall i \in I, j \in J, h \in H \quad (12)$$

$$Z_{hj} = \{0, 1\}, \forall h \in H, j \in J \quad (13)$$

Objective functions (1)-(2) aim to minimize the cycle time and total ergonomic risks in all workstations respectively. Constraint (3) ensures that the total processing time of each workstation does not exceed the cycle time. Constraint (4) ensures that each task is allocated to exactly one workstation either at the entrance or exit subline and carried out by one worker. Constraint (5) indicates that each workstation needs to be allocated at least one task. Constraints (6)-(7) guarantee that the precedence relationship must be satisfied when tasks are allocated into workstations. When assigning workers to workstations, it should be ensured that the number of workers is equal to that of workstations. Constraint (8) implies that one task can be processed by the h-th worker in the j-th workstation only when the h-th worker is assigned to that workstation. Constraint (9) ensures that each workstation is allocated to one worker and constraint (10) ensures that each worker is allocated to one workstation. Constraints (11)-(13) again define the binary nature of the decision variables.

3.4. OCRA method

Since each workstation has one worker, this paper utilizes ER_j to represent the ergonomic risks of the worker assigned to workstation j and then employs the OCRA method proposed by Occhipinti (1998) to calculate them. This method comprehensively evaluates the ergonomics of posture, force, repetitiveness and additional risks, and is presented as follows:

$$ER_j = \text{OCRA index} = \frac{\text{Actual frequency}}{\text{Recommended frequency}} \quad (14)$$

$$\text{Actual frequency} = \frac{\text{Number of technical actions within a cycle}}{cycle\ time} \times 60 \quad (15)$$

$$\text{Recommended frequency} = OS \times PM \times FoM \times RM \times AdM \quad (16)$$

Here OS is the constant of organization specific parameters and it is assumed to be 18 in an ideal condition according to Occhipinti (1998). PM is a multiplier related to awkward postures when workers execute tasks. It mainly evaluates two types of postures: severe and mild postures, and its value is determined by the duration ratio to cycle time denoted in Table 1. If more than one posture exists in one workstation, the lowest value over all postures is assigned to PM. FoM is a multiplier that evaluates the physical effort when workers execute technical actions. This multiplier is a function of the average force level within workstation shown in Table 2. RM is a multiplier for repetitiveness when repetitive tasks are executed at high frequency. This multiplier is related with the cycle time and the same technical actions of upper limbs. If the cycle time is less than 15s and/or the duration of same actions exceeds 50% of cycle time, it takes value 0.7; otherwise it equals to 1.0. AdM is a multiplier for additional risk factors such as exposure to cold or hot, noise, vibrating tools and lightening, etc. The value of AdM depends on the exposure duration in these additional risks and is shown in Table 1.

Table 1. Posture and additional risk multiples related to cycle time percent

Cycle time percent	≤1/4	1/3	1/2	2/3	3/3
PM for severe postures	1.0	0.7	0.65	0.6	0.5
PM for mild postures	1.0	1.0	1.0	0.7	0.6
AdM	1.0	0.95	0.925	0.9	0.8

Table 2. Force multiple related to different force level

Average force level	5%	10%	20%	30%	40%	≥ 50%
FoM	1	0.85	0.65	0.35	0.2	0.01

3.5. A illustrative example

A illustrative example is presented to give a better insight of this new problem. The precedence graph is given in Figure 1, and the processing time and ergonomic risk parameters of this example are shown in Tables 3-4. In Table 4, column 2 means the frequency actions per minute of a task, so that the number of actions during this task can be calculated by frequency actions×processing time/60. The rest of the columns represent the duration ratio to task time of average force, additional factor, repetitive actions and postures respectively. Note that two types of repetitive actions, three types of severe and mild postures are involved in this example. It is assumed that the given number of workstations and that of workers are equal to 3. One of task assignment plans is illustrated in Figure 1 where workstations 1-3 are allocated with workers 3, 1 and 2 respectively.

Table 3. Processing times and input data for the example tasks.

Task	Processing time/s				*NST*	*TST*	*NPT*	*TPT*
	Worker 1	Worker 2	Worker 3	Average*				
1	55	67	73	65	8	575	0	0
2	48	56	75	60	3	272	1	65
3	75	93	97	88	3	272	1	65
4	36	53	38	42	2	113	1	65
5	40	54	44	46	2	226	3	213
6	111	45	41	66	1	47	2	107
7	47	44	49	47	0	0	3	173
8	157	133	86	125	1	101	4	259
9	149	58	97	101	0	0	5	384

* rounded to the nearest integer

Table 4. Ergonomic risk parameters for the example tasks

Task	Frequency actions	Average force	Additional factor	Repetitive actions		Severe postures			Mild postures		
				A	B	1	2	3	1	2	3
1	5	0.200000	0.607110	0.000000	0.000000	0.738421	0.000000	0.000000	0.000000	0.686778	0.530845
2	19	0.050000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.557214	0.612380	0.519493
3	10	0.100000	0.000000	0.442922	0.000000	0.000000	0.753378	0.579502	0.000000	0.000000	0.000000
4	2	0.100000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.668380	0.000000
5	4	0.200000	0.000000	0.000000	0.722840	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
6	7	0.050000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.554896	0.000000	0.000000
7	8	0.200000	0.000000	0.000000	0.000000	0.594419	0.000000	0.000000	0.000000	0.510333	0.000000
8	16	0.100000	0.000000	0.000000	0.000000	0.000000	0.580080	0.000000	0.787345	0.000000	0.572485
9	2	0.100000	0.000000	0.000000	0.000000	0.574225	0.770713	0.716622	0.000000	0.000000	0.000000

In this task and worker assignment plan, the total processing time of workstations are equal to 219, 234 and 240 respectively, and hence the cycle time is 240. Besides, the ergonomic risk values of workstations calculated by the OCRA method equal to 0.544070, 0.609623 and 1.035197 respectively, and the total ergonomic risk is 2.18889. Specifically, take workstation 1 for example, there are three tasks 1, 7 and 9, and the corresponding processing times by worker 3 are respectively

73, 49 and 97. The number of frequency actions of these tasks are respectively 7 ([5×73/60]), 7 ([8×49/60]) and 4 ([2×97/60]). Hence the actual frequency in workstation 1 is (7+7+4)×60/240=4.5. The workload of the first posture at workstation 1 is 138.7311s (0.738421×73+0.594419×49+0.574225×97) for 57.8046% (138.7311/240) of the cycle time and its corresponding *PM* is 0.6. The *PM*s of the rest postures are depicted in Table 5. The *PM* with the lowest value 0.6 is set as the posture multiplier at workstation 1. Further, the time-weighted average force within workstation 1 is 14.2083% ((0.2×73+0.1×49+0.2×97)/240) of the cycle time, and hence *FoM* is 0.765833 (by interpolation). The durations of the two types of repetitive actions are both 0 and the cycle time is more than 15s, so that *RM* is equal to 1.0. The duration of the additional factors is 44.31903s (73×0.60711) for 18.4662% of cycle time, so that *AdM* is 1.0. Thus, the ergonomic risk value at workstation 1 is 4.5/(18×0.6×0.765833×1.0×1.0)=0.54407. With the same OCRA method, the ergonomic risk values of workstations 2 and 3 are calculated and shown in Table 6.

Table 5. The *PM* values of all awkward postures

	Severe postures			Mild postures		
	1	2	3	1	2	3
Duration	138.7311	74.75916	69.51233	0.00000	75.14111	38.75169
% cycle time	0.578046	0.311497	0.289635	0.00000	0.313088	0.161465
PM	0.6	0.7	0.7	1.0	1.0	1.0

Table 6. Ergonomic risk values of all workstations

workstation	Total number of actions	Actual frequency	*PM*	*FoM*	*RM*	*AdM*	*Eg*
1	18	4.5	0.6	0.765833	1.0	1.0	0.544070
2	42	10.5	1.0	0.956875	1.0	1.0	0.609623
3	42	10.5	0.7	0.805000	1.0	1.0	1.035197

4. Restarted Iterated Pareto Greedy algorithm

The iterated greedy (IG) is a simple local search-based method which has been successfully applied to discrete and combinational optimization problems. The main feature of the IG is its simplicity, in the sense that this algorithm does not need to embed specific knowledge and has few control parameters (Ruiz and Stutzle, 2007). We first introduce the original IG method and later the required adaptations for the U-shaped assembly line worker assignment and balancing problem with ergonomic risks.

4.1. Original iterated greedy algorithm

The original IG starts with an initial solution usually generated by heuristic rules, such as the NEH heuristic in flow shop scheduling. In the main IG loop, the main phases are destruction and construction. Destruction extracts some elements from the solution and these extracted elements are reinserted back into different positions in this solution to constitute a new solution in the construction phase. Additionally, a local search procedure is applied to improve the new reconstructed solution. Finally, an acceptance criterion is used to determine whether the incumbent solution should be replaced by the new solution. The pseudo-code of the original iterated greedy

algorithm is shown in Figure 2.

Algorithm: Iterated Greedy Algorithm

1: **Input**: An initial solution π
2: **Output**: Best found solution π^*
3: **Begin**:
4: $\pi^*=\pi$;
5: **While** termination criterion not satisfied **do**
6: π_1=Destruction(π);
7: π_2=Construction(π_1);
8: π_3=LocalSearch(π_2);
9: π=AcceptanceCriterion(π, π_3);
10: If π_3 is better than π^* **then**
11: $\pi^*=\pi_3$;
12 **End if**
13: **End while**
14: **Return** π^*

Fig. 2. Pseudo-code of the original iterated greedy algorithm (Ruiz and Stutzle, 2007).

The IG was first proposed by Ruiz and Stutzle (2007), and has been mainly applied to different scheduling problems. Some examples are the simple iterated greedy algorithm (Tasgetiren et al., 2017), improved iterated greedy algorithm (Ding et al., 2015; Pan et al., 2007) and effective iterated greedy algorithm (Pan and Ruiz, 2014). For multiple objectives, Minella et al. (2011) extended the IG to more than one objective in order to search for Pareto dominant solutions. They referred to the extended algorithm as the RIPG (Restarted Iterated Pareto Greedy). However, to our knowledge, there is only one paper employing a simple IG for single objective type-I two-sided assembly line balancing problems by Li, Z.X. et al. (2017). Therefore, in order to apply the IG algorithm to the multi-objective UALWABP, an improved Restarted Iterated Pareto Greedy algorithm (RIPG) is developed here to minimize cycle times and ergonomic risks simultaneously. The proposed RIPG starts with an initial solution improved by heuristic rules and this solution is stored in an external Pareto Archive (*POS*). While the termination criterion is not satisfied, the RIPG successively implements a precedence-based greedy, local search phases, acceptance criterion and a restart mechanism. All these steps are detailed in the following sections.

4.2. Problem-specific and heuristic-based initialization

For the UALWABP, the proposed RIPG starts with an initial solution that contains two vectors: worker assignment and task allocation. For worker assignment, the length is equal to *NH* and the elements represent the worker allocated to each workstation. For example, if there are three workers and the corresponding worker assignment vector is [3, 1, 2], it means that workstations 1-3 are assigned to workers 3, 1 and 2 respectively.

For the task allocation, it is well known in the UALWABP that heuristics can find good initial solutions, especially on large instances. Therefore, we adopt some high performing heuristic rules designed by Baykasoğlu and Özbakır (2006). According to Rabbani et al. (2012), the task allocation vector length is equal to *NI* and each value represents the heuristic rule number used to allocate each task. The values are randomly generated between [1, 10] for each position (10 different heuristic rules are employed). Note that repetitions are allowed. These 10 values come from 10 of the best

rules presented in Baykasoğlu and Özbakır (2006), i.e., 1: Shortest Processing Time, 2: Longest Processing Time, 3: Smallest total number of successors, 4: Largest total number of successors, 5: Largest total time of successors, 6: Smallest total time of successors, 7: Smallest total number of predecessors, 8: Largest total number of predecessors, 9: Largest total time of predecessors and 10: Smallest total time of predecessors.

After obtaining the task allocation vector, the task allocation order is determined by a combination of the precedence relationships and these heuristic rules. Note that the decoding, precedence-based greedy and local search phases are executed based on this task allocation order. The process for determining the task allocation order is as follows:

Step 1: Let X be the first value in the current task allocation vector (heuristic rule number).

Step 2: Based on the precedence relationships, calculate the current optional task candidate sets B_1 and B_2, where B_1 stores the tasks which can be allocated to the entrance of the workstation and B_2 stores the tasks that can be allocated to the exit. In other words, if the immediate predecessors of a task have all been allocated, this task can be stored in B_1. Otherwise, it can be put into B_2 when the immediate successors of this task have all been allocated.

Step 3: Select the task with the highest priority from the union of the two current candidate sets B_1 and B_2 according to the application of heuristic rule X, denote it as Y and put it in bold if it is from B_2.

Step 4: Place Y in the last position in the task allocation order and remove it from the candidate sets.

Step 5: Remove X from the task allocation vector. If all tasks have not been allocated, return to *Step* 1. Otherwise, terminate the process.

Take the precedence relationships in Fig. 1(a) as example data to illustrate this process. The corresponding processing times and other input data are shown in Table 3. In this table, *NST* and *NPT* mean the number of successor and predecessor tasks respectively. *TST* and *TPT* are the total average processing times of successor and predecessor tasks respectively. These values can be calculated based on Fig. 1(a) considering the average times of each task (rounded to the nearest integer). Table 7 presents an example of the complete process. The initial task allocation vector is [4, 1, 5, 9, 4, 1, 8, 6, 3], and according to the above steps, a task allocation order [1, 4, 2, **9**, 3, 5, 6, **7**, **8**] is obtained. The tasks in bold are those allocated to the exit of the workstations.

Table 7. Step by step application of the task allocation order for the example.

Initial Task allocation vector	*Step* 1	*Step* 2	*Step* 3	*Step* 4 (task allocation order)
[4,1,5,9,4,1,8,6,3]	X=4	B_1=[1], B_2=[7,9]	Y=1	[1]
[1,5,9,4,1,8,6,3]	X=1	B_1=[2,3,4], B_2=[7,9]	Y=4	[1,4]
[5,9,4,1,8,6,3]	X=5	B_1=[2,3,6] , B_2=[7,9]	Y=2	[1,4,2]
[9,4,1,8,6,3]	X=9	B_1=[3,6], B_2=[7,9]	Y=9	[1,4,2,**9**]
[4,1,8,6,3]	X=4	B_1=[3,6], B_2=[7,8]	Y=3	[1,4,2,**9**,3]
[1,8,6,3]	X=1	B_1=[5,6], B_2=[7,8]	Y=5	[1,4,2,**9**,3,5]
[8,6,3]	X=8	B_1=[6,8], B_2=[7,8]	Y=6	[1,4,2,**9**,3,5,6]
[6,3]	X=6	B_1=[7,8], B_2=[7,8]	Y=7	[1,4,2,**9**,3,5,6,**7**]
[3]	X=3	B_1=[8], B_2=[8]	Y=8	[1,4,2,**9**,3,5,6,**7**,**8**]

4.3. Decoding scheme

The above process obtains the task allocation order and worker assignment only. A detailed allocation of all the tasks to each workstation and the values of the objective functions are

determined with a decoding scheme similar to that proposed by Scholl (1999) and Zhang et al. (2018). According the survey reported by Scholl (1999), this decoding scheme is an iterated search procedure that solves the problem with a monotonically increasing cycle time for a given number of stations. Compared to the binary search and upper bound search decoding schemes, this decoding scheme can guarantee the smallest cycle time.

Step 1: Calculate the initial cycle time CT_0 by $CT_0 = \left\lceil \sum_{i=1}^{NI} \min_{1 \le h \le NH} t_{ih}/NW \right\rceil$ and set $CT_1 = CT_0$.

Step 2: According to the sequence of the task allocation order, select the first task *i* placed at the *i*th position in this sequence.

Step 3: Create a new workstation *wc* and set the workstation time WT_{wc}=0.

Step 4: Determine the processing time t_{ih} of task *i*, which is operated by worker *h* assigned to the current workstation.

Step 5: If the total of WT_{wc} and t_i does not exceed the cycle time, allocate task *i* to current workstation *wc* and set WT_{wc}=WT_{wc}+t_i, *i*=*i*+1; Otherwise, if the number of workstations is less than *NW*, return to *Step* 3; otherwise go to *Step* 7.

Step 6: If *i*>*NI*, go to *Step* 8; otherwise, return to *Step* 4.

Step 7: Set CT_1= CT_1+1 and return to *Step* 2 to reallocate tasks.

Step 8: Calculate the cycle time and total ergonomic risk of the current feasible solution.

The computational complexity of this decoding is $O(NI \times (CT - CT_0))$, which is the same as the binary search and upper bound search decoding schemes. We provide in Table 8 a detailed example of the application of the procedure.

Table 8. A solution decoding procedure example.

Step 2	*Step* 3	*Step* 4	*Step* 5	*Step* 6	*Step* 7
Task 1	*wc*=1, WT_1=0	t_1=73	Wc_1=[1], WT_1=73, task 4	Go to *Step* 4	
		t_4=38	Wc_1=[1,4], WT_1=111, task 2	Go to *Step* 4	
		t_2=75	*Go to Step* 3		
	wc=2, WT_2=0	t_2=48	Wc_2=[2], WT_2=48, task 9	Go to *Step* 4	
		t_9=149	*Go to Step* 3		
	wc=3, WT_3=0	t_9=58	Wc_3=[9], WT_3=58, task 3	Go to *Step* 4	
		t_3=93	Wc_3=[9,3], WT_3=151, task 5	Go to *Step* 4	
		t_5=54	*Go to Step* 7		CT_1= CT_1+1, Go to *Step* 2
.	.	.	.	.	CT_1=264, Go to *Step* 2
.	.	.	.	.	
.	.	.	.	.	
Task 1	*wc*=1, WT_1=0	t_1=73	Wc_1=[1], WT_1=73, task 4	Go to *Step* 4	
		t_4=38	Wc_1=[1,4], WT_1=111, task 2	Go to *Step* 4	
		t_2=75	Wc_1=[1,4,2], WT_1=186, task 9	Go to *Step* 4	
		t_9=97	*Go to Step* 3		
	wc=2, WT_2=0	t_9=149	Wc_2=[9], WT_2=149, task 3	Go to *Step* 4	
		t_3=75	Wc_2=[9,3], WT_2=224, task 5	Go to *Step* 4	
		t_5=40	Wc_2=[9,3,5], WT_2=264, task 6	Go to *Step* 4	
		t_6=111	*Go to Step* 3		

wc=3, WT_3=0	t_6=45	Wc_3=[6], WT_3=45, task 7	Go to *Step* 4
	t_7=44	Wc_3=[6,7], WT_3=89, task 8	Go to *Step* 4
	t_8=133	Wc_3=[6,7,8], WT_3=222	Go to *Step* 8

4.4. Precedence-based greedy phase

The greedy phase consists of two key parts in the proposed RIPG algorithm: destruction and construction. Since there are two sub-problems, we first execute the greedy phase on the worker assignment vector, and then on the encoded task allocation vector.

For the worker assignment vector, $d_1\%$ of the NH workers are extracted and then reinserted into all possible positions to generate a set of partial solutions.

For the encoded task allocation vector, a consecutive block of $d_2\%$ of the NI tasks are randomly extracted from the incumbent solution in the destruction phase and then are reinserted into all possible positions to generate a set of partial solutions in the construction phase. It should be noted that after reinsertion, the positions of the tasks must be different from their original positions and should satisfy the precedence constraints.

A temporary set is used to store non-dominated solutions in the current solution from the set of partial solutions and a new solution is randomly selected from this temporary set for the precedence-based local search phase (to be explained in the next section). This is similar to the reconstruction procedure in the RIPG proposed by Minella et al. (2011).

Note that the number of the extracted workers (d_1) and tasks (d_2) might exert a great influence on the performance of the algorithm. Large values of d_1 and d_2 destroy the current solution too much, adding excessive diversification and resulting in a random walk, whereas small values make it difficult to escape from local optima. Thus, the values of d_1 and d_2 need to be carefully calibrated. This will be carried out in the following sections.

4.5. precedence-based local search phase

Two local search operators are proposed, each dealing with one of the two sub-problems. Similarly to the greedy phase, we first employ worker swap to search the neighboring solutions around the worker assignment space, and then carry out a deep local search on the task allocation space.

For worker swap, we swap all possible pairs of workers until a local optima solution is found, i.e., when a better swap is found all possible worker swaps are analyzed again. More specifically, a worker is randomly selected and swapped with all other workers to generate a set of partial solutions. If improvements are found, the process is repeated until all workers have been swapped with no improvements.

For the task allocation local search, all tasks from the greedy solution are tested in all positions meeting the precedence relations to generate a set of solutions. Similarly to the worker swap, if improvements are found, the process is repeated.

Through the above local search phase, a set of solutions is obtained. In this set, these non-dominated solutions are stored in the temporary set and the remaining dominated remaining solutions are removed.

4.6. Acceptance criterion

The proposed acceptance criterion operator consists of two steps: First, the external Pareto

Archive (*POS*) needs to be updated. Each one of the solutions in the temporary set is compared with the solutions in the *POS* and are included into it if found to be non-dominated by the solutions in the *POS*. Similarly, solutions in the *POS* which are dominated by the newly added solutions are removed.

The second step is to decide whether any new solution in the temporary set should replace the incumbent solution in the RIPG. This paper employs the acceptance criterion proposed by Hatami et al. (2015). If there are some solutions in the temporary set not dominated by the solutions in the *POS*, one solution randomly selected from these newly non-dominated solutions is accepted as a new incumbent. Otherwise, each solution π_2 in the temporary set is accepted with two probabilities $e^{(-RPD1)}$ and $e^{(-RPD2)}$, where $RPD1=(CT(\pi_2)-CT(\pi_0))/CT(\pi_0) \times 100\%$, $RPD2=(TER(\pi_2)-TER(\pi_0))/TER(\pi_0)\times 100\%$, and if more than one solution is accepted, one solution is randomly selected to replace the incumbent solution.

4.7. Restart phase

To enhance diversification, a restart mechanism proposed by Minella et al. (2011) is employed in the proposed RIPG. If there are no non-dominated solutions generated at any given iteration, a counter *dn* is increased. When the counter *dn* is larger than a given restart number *DN*, a solution will be selected from the *POS* by the restart mechanism. This method sets a *select_counter* which is the number of times a solution has been selected already. In order to avoid selecting solutions repeatedly, the crowding distance defined by Deb et al. (2002) divides the *select_counter* to obtain a modified crowding distance. The solution with the largest value in the modified crowding distance is selected. The proposed restart mechanism is depicted in Fig. 3. Note that the restart number *DN* is the last parameter to calibrate.

Mechanism: restart mechanism

Input: Pareto front set *POS*
Output: The selected solution π
POS_size=|*POS*|; *M_count*=Number of objectives;
Begin:
  For *m*=1 to *M_count* **do**
    Sort *POS* using objective *m*;
    Calculate the maximum and minimum objective values maxf_m, minf_m;
    For *i*=2 to *POS_size*-1 **do**
      $POS[i]_{dist}= POS[i]_{dist}+(POS[i+1]_{dist} - POS[i-1]_{dist})/(\max f_m - \min f_m)$;
    End for
    $POS[1]_{dist}= POS[1]_{dist}+1/m$;
    $POS[POS\_size]_{dist}= POS[POS\_size]_{dist}+1/m$;
  End for
  For *i*=1 to *POS_size* **do**
    $POS[i]_{dist}= POS[i]_{dist}/ POS[i]_{select\_counter}$;
  End for
  π =the solution with largest distance;
Return π

Fig. 3. Proposed restart procedure.

Fig. 4 shows the pseudocode of the proposed RIPG algorithm with all the above phases.

Algorithm: Restarted Iterated Pareto Greedy

1: **Input**: An initial solution π
2: **Output**: Pareto front set *POS*
3: **Begin**:
4: Generate an initial solution π by the heuristic rules;
5: Store π in *POS*;
6: **While** termination criterion not satisfied **do**
7: Extract d_1% workers and d_2% tasks from π in the destruction phase;
8: Generate a set S_C of solutions by reinserting the extracted workers and tasks in the construction;
9: Set a temporary set to store the non-dominated solutions of S_C;
10: Randomly select a solution π_1 from the temporary set;
11: Generate a set S_L of solutions around π_1 in the local search;
12: Update the temporary set by the set S_L;
13: Update the *POS* by the temporary set;
14: π=AcceptanceCriterion(π, temporary set);
15: **If** the *POS* is not updated for *DN* times **then**
16: π=restart(*POS*);
17: **End If**
13: **End while**
14: **Return** *POS*

Fig. 4. Pseudocode of the proposed RIPG algorithm.

5. Computational results and discussion

We have carried out three sets of experiments to measure the optimization of the proposed RIPG algorithm. First, a calibration experiment is conducted with a view to making a decision on the best parameter combination for the RIPG as well as for other compared algorithms. Second, we compare the RIPG with nine other well-known multi-objective algorithms. Finally, a graphical methodology, the Differential Empirical Attainment Function (Diff-EAF) (López-Ibáñez et al., 2006a), is employed to show the differences between the Empirical Attainment Functions (EAFs) obtained by the RIPG and other algorithms. All algorithms are coded in the C++ programming language and are run on a computer with Intel(R) Core(TM) i5 4440 processor running at 2.80GHz and with 2.00GBytes of main RAM memory.

In this paper, the non-dominated solutions in the combination of the results obtained by all the compared algorithms are regarded as the best-known approximation of the true Pareto frontier. Two Pareto-compliant performance indicators are used to evaluate the Pareto front set algorithms obtained algorithms in the experiments. The first indicator is the HyperVolume Ratio *HVR* proposed by Tan et al. (2006), which is the ratio between the hypervolume of the obtained Pareto set and that of the best known approximation of this frontier. This indicator is calculated by equation (25). In the *HVR*, n and m represent the number of the Pareto solutions obtained and that of the objectives respectively. v_i refers to the ith hypercube, whose diagonal corners are the objective vector of solution i in the obtained Pareto set and that of reference point W. Among them, reference point W is constructed with a vector of the worst objective values. So, the hypervolume is the union of all

hypercubes. According to the research by Tan et al. (2006), since the *HVR* indicator combines both the distance and the spread of solutions, it is capable of serving as the most appropriate scalar indicator. Pareto sets with *HVR* values closer to 1 indicate better approximations to the true frontier. Another indicator is the Unary Epsilon Indicator I_ε proposed by Knowles et al. (2006), which measures the minimum distance between an obtained Pareto front set and the true frontier or the best known approximation of this frontier. This indicator is calculated using equation (26). In I_ε, S is an obtained Pareto front set and P is the true frontier or the best-known approximation. χ^1 and χ^2 are the solutions of S and P, respectively, and f_j indicates the jth objective function. The obtained Pareto front set with I_ε closer to 1 indicates that this front is close to the true Pareto frontier or to its best approximation.

$$HVR = \frac{volume(\cup_{i=1}^{n} v_i)}{volume(\cup_{j=1}^{m} v_j)} \tag{25}$$

$$I_\varepsilon = I_\varepsilon(S,P) = \max_{\chi^2} \min_{\chi^1} \max_{j} \frac{f_j(\chi^1)}{f_j(\chi^2)} \tag{26}$$

We use the well-known benchmark data set for the UALBP-II proposed by Talbot et al. (1986), Hoffmann (1990) and Scholl (1995). The benchmark is organed into 17 groups, specifically including Buxey, Sawyer, Lutz1, Gunther, Kilbridge, Hahn, Warnecke, Tonge, Wee-Mag, Arcus1, Lutz2, Lutz3, Mukherje, Arcus2, Barthold, Barthol2 and Scholl families, for a total 302 instances. According to the method proposed by Miralles et al. (2008), the processing time of task i by worker h is randomly generated between $[0, t_i]$, in which t_i is the original processing time of task i. Since there are 27 replications of ergonomic parameters for each instance of simple assembly line balancing problem proposed by Otto and Scholl (2011), we randomly select 3 different ergonomic parameter sets for each test instance. Hence, a total 906 instances are employed in this section to test the performance of proposed algorithm.

5.1. Calibration of the proposed RIPG

Before testing the performance of our proposed RIPG, the Design of Experiments (DOE) technique is employed to select the best parameter configuration. A full factorial design of experiments is used to explore the combinations of parameters. Results are analyzed by means of the multifactor Analysis of Variance (ANOVA). The ANOVA is an important parametric statistical inference tool used to check the three main hypotheses: normality, homoscedasticity and independence of the residuals. For the RIPG algorithm, there are four factors to calibrate: (1) The removed ratio of workers d_1 at four levels: 10, 30, 50 and 70%. (2) The removed ratio of tasks d_2 at nine levels: 1, 3, 5, 10, 15, 20, 25, 30 and 35%. (3) The number of iterations before restart DN at four levels: no restart, restart with DN =5, 15 and 20. (4) The precedence-based local search phase, tested at two levels, used and not used (ON/OFF). Through the full factorial design, there are a total of 4·9·4·2=288 experiment configurations and each configuration is run with all 10 calibration instances. These are different to the final 906 testing instances but are generated using the same procedure. Each instance is solved 10 times to obtain 10 different Pareto sets. As a result, a total of 28,800 experiments are carried out. The stopping criteria for all algorithm configurations is set to a CPU time limit of $NI \times \rho$ milliseconds, where ρ=5 is set for these experiments. The average *HVR* and I_ε of each experiment configuration is regarded as the final response value. The means plots of *HVR* and I_ε with Tukey's Honest Significant Difference (HSD) 95% confidence intervals for the calibration of the proposed RIPG are shown in Figs.5 and 6. From the figures it is clear that the

restart and local search operators contribute to the algorithm (DN=0 and LS=OFF result in statistically worse performance). From the analysis we conclude that the best parameter combination is d_1=50, d_2=5, DN=5 and the precedence-based local search.

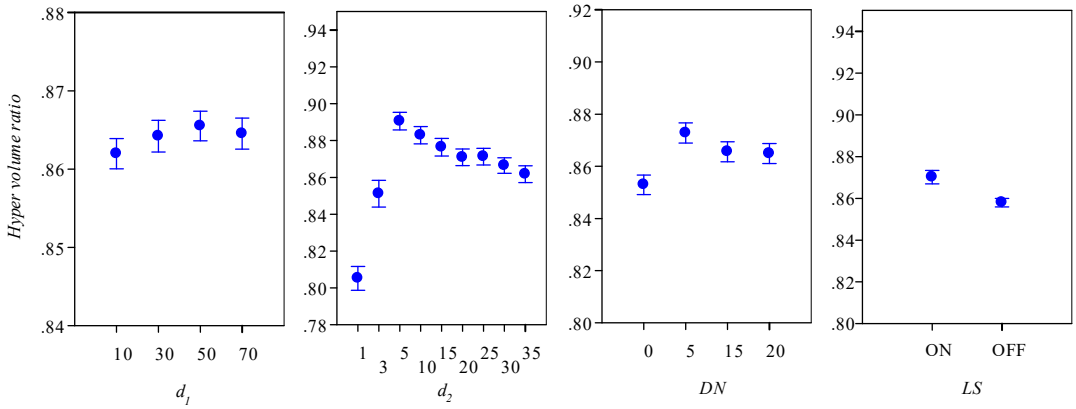


Fig. 5. Means plots of *HVR* with Tukey's Honest Significant Difference (HSD) 95% confidence intervals for all the factors in the ANOVA calibration experiment for the proposed RIPG.

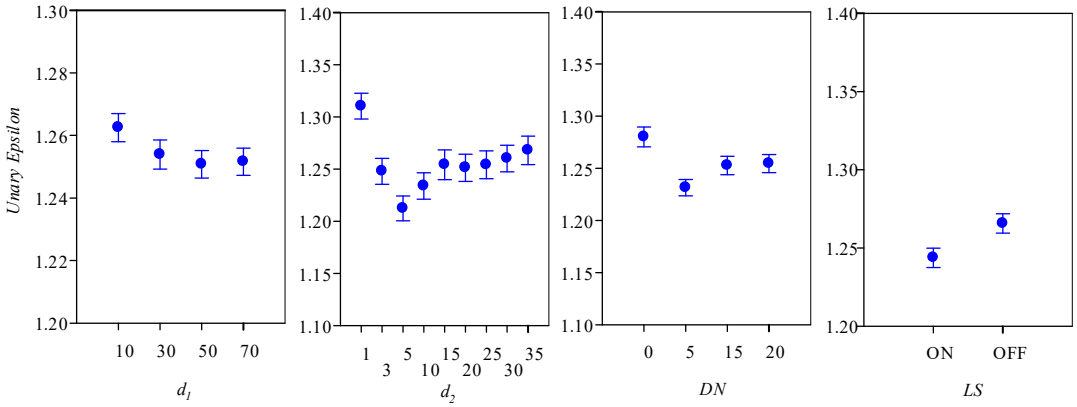


Fig. 6. Means plots of I_ε with Tukey's Honest Significant Difference (HSD) 95% confidence intervals for all the factors in the ANOVA calibration experiment for the proposed RIPG.

To obtain a convincing comparison, the parameters of the nine compared algorithms are also calibrated using the same procedure. The compared algorithms are ICA, MABC, MOEA1-5, MOPSO and MOSA (see Table 9), and are chosen due to them performing well on related assembly line balancing problems. For the calibration of the compared algorithms, a total of 22,500 experiments were carried out. The details of the calibration are given as on-line materials.

Table 9. Summary of the compared algorithms.

Algorithm	Acronym	Description
Imperialist competitive algorithm (Nourmohammadi et al., 2013)	ICA	Shares objective functions in Sect.3.
Multi-objective artificial bee colony (Saif et al., 2017)	MABC	Shares the code and encode in Sect.4.2.1.
Multi-objective evolutionary algorithm (Zacharia and Nearchou, 2016)	MOEA1-5	MOEA1-5s use their own fitness functions (1-5).
Multi-objective Particle swarm	MOPSO	Repository size (*Nr*) is not limited, and the personal learning

optimization (Rabbani et al., 2016)		coefficient and the global learning coefficient are randomly generated between [0, 1].
Multi-objective Simulated annealing (Baykasoglu, 2006)	MOSA	Add the worker swap for neighbor structure.

5.2. Computational comparisons with state-of-the-art methods

We now compare our proposed RIPG algorithm with the above mentioned nine high performing algorithms. We test a bi-objective combination with cycle time and ergonomic risk for these algorithms. All these algorithms are re-coded in the same computer language and run on the same computer for a fair comparison (the same computer as in the calibration). Two stopping criteria with CPU time limits of $NI \times \rho$ milliseconds (ρ=10 and 20) are set for the experiments. Each algorithm is run on each one of the 906 test instances with the two stopping criteria 10 different times. A total of 181,200 experiments are collected and analyzed by the ANOVA technique.

Table 10 demonstrates the average values of *HVR* and I_ε for the nine compared algorithms and the RIPG for the two stopping criteria. From this table we can clearly observe that our proposed RIPG algorithm obtains better results than all the other nine methods for both *HVR* and I_ε at the two stopping criteria. Apart from the RIPG, the ICA and MOPSO algorithms demonstrate the second-best and third-best performance for both *HVR* and I_ε. As for MOEA1-5, these algorithms perform similarly which means the different fitness functions in MOEA have little impact on the bi-objective UALWABP. MABC and MOSA obtain the worst results. All these analyses are based on the average values of 906 instances. For some specific instances, MOEA may perform better. For example, on the large instance with 297 tasks and 29/38/50 workers, the values of *HVR* obtained by MOEA1-5 are not less than 0.80, which is close to that of ICA and MOPSO. Thus, we can conclude that the proposed RIPG algorithm clearly performs better on average.

Furthermore, Figs.7 and 8 show the means plots after the ANOVA statistical tests with 95% confidence level Honest Significant Difference (HSD) intervals for the *HVR* and I_ε of the compared algorithms and the RIPG for ρ=10 and 20. One observation we can obtain from these results is that the proposed RIPG algorithm outperforms all the other nine methods for both *HVR* and I_ε at the two stopping criterions while MABC has the worst performance on average.

Table 10. Computational results for the compared algorithms and the proposed RIPG.

No.	Algorithm	ρ=10		ρ=20	
		HVR	I_ε	*HVR*	I_ε
1	ICA	0.833135	10.00574	0.835386	9.874739
2	MABC	0.776917	11.19130	0.795817	10.62969
3	MOEA1	0.795915	10.68218	0.803961	10.39131
4	MOEA2	0.795479	10.68884	0.803434	10.4049
5	MOEA3	0.794753	10.70137	0.802883	10.41035
6	MOEA4	0.795709	10.68096	0.803792	10.39182
7	MOEA5	0.800629	10.62496	0.811749	10.28596
8	MOPSO	0.817560	10.28869	0.822529	10.03679
9	MOSA	0.778282	11.15792	0.776830	11.09135
10	RIPG	**0.840798**	**2.282513**	**0.838290**	***2.256196***

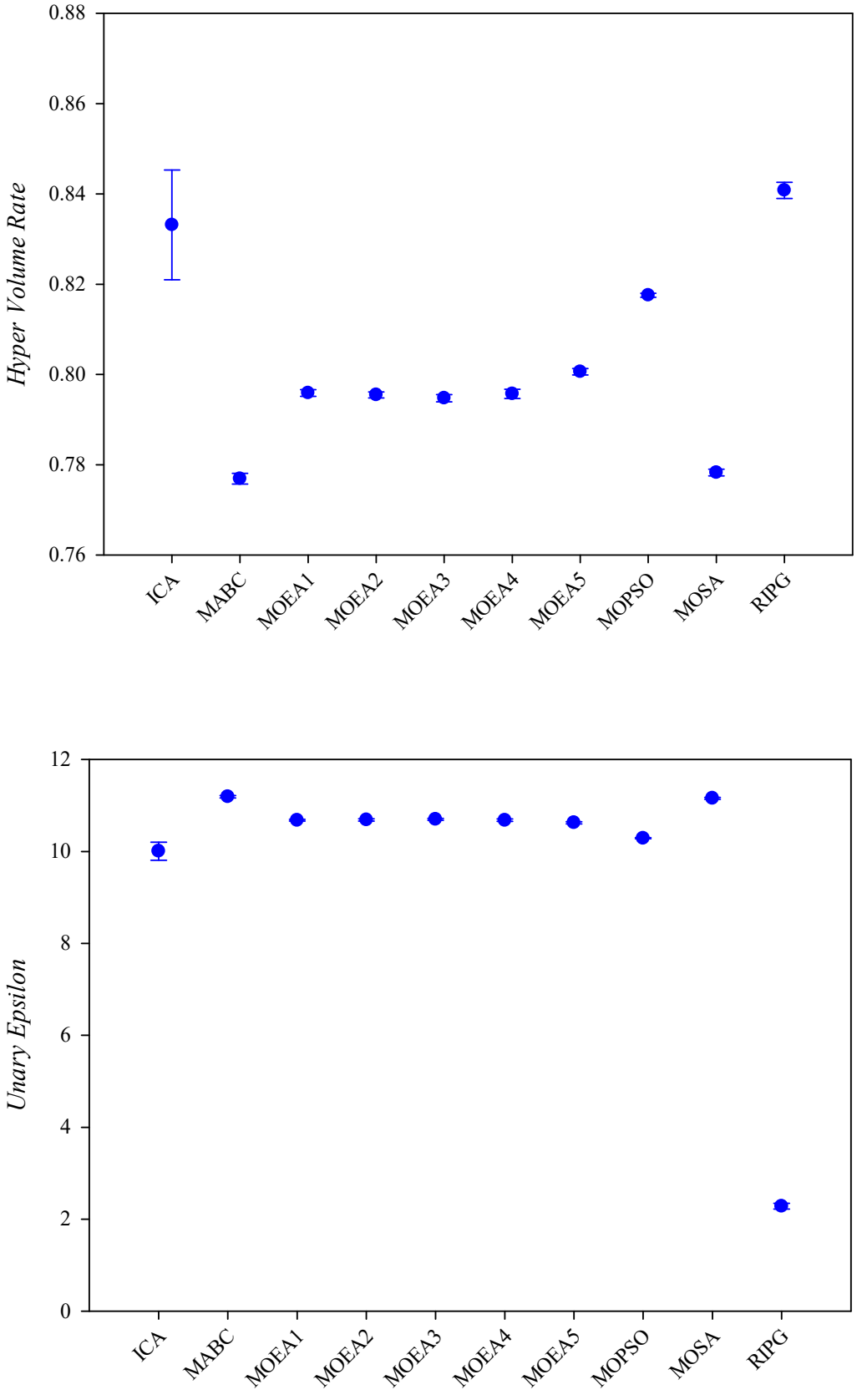
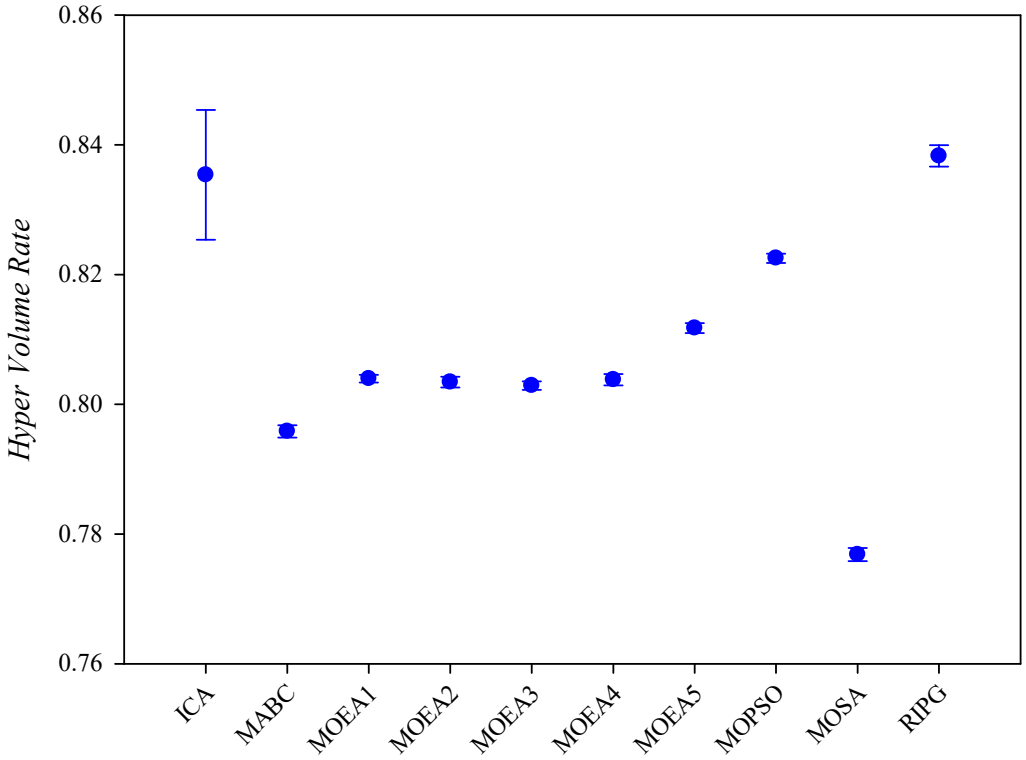


Fig. 7. Means plots of *HVR* and I_ε with Tukey's Honest Significant Difference (HSD) 95% confidence intervals for

all the compared algorithms and the proposed RIPG and ρ=10 stopping criterion.



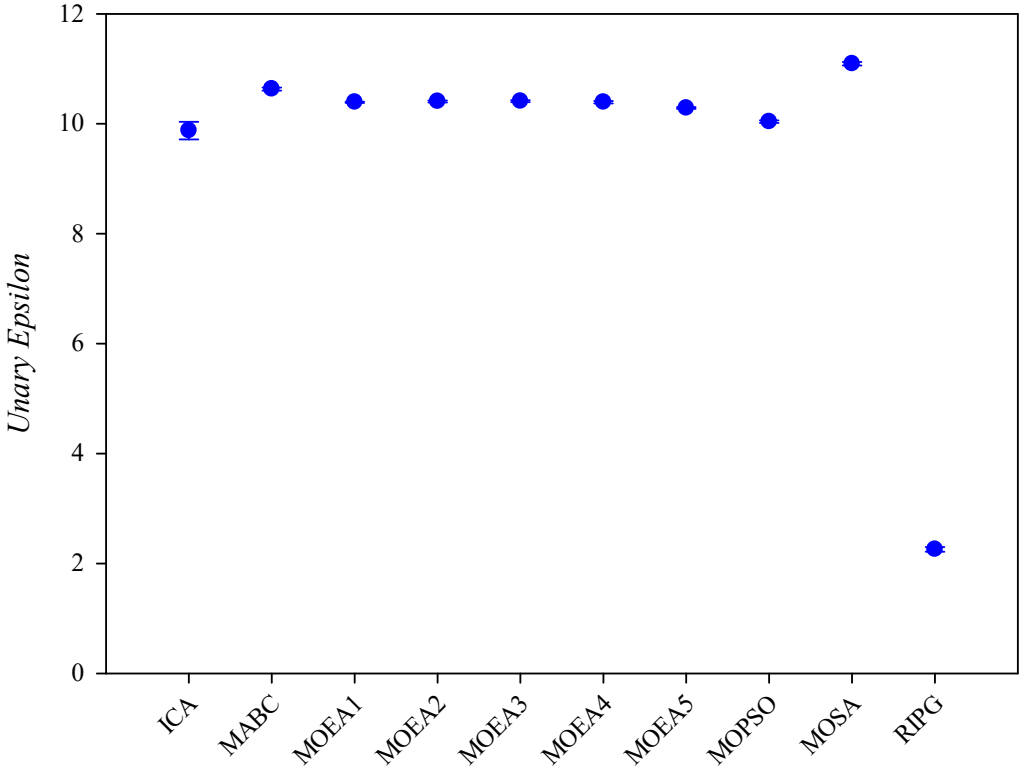


Fig. 8. Means plots of *HVR* and I_ε with Tukey's Honest Significant Difference (HSD) 95% confidence intervals for all the compared algorithms and the proposed RIPG and $\rho=20$ stopping criterion.

5.3. Differential Empirical Attainment Functions

Unary scalar indicators cannot infer the information about the spatial behavior of algorithms. As a result, it is not possible to observe whether a given algorithm performs better in a given part of the objective space. Hence, the empirical attainment function (EAF) proposed by Grunert da Fonseca et al. (2001) is used to deal with this issue. EAF shows the probability of a random point in the objective space being dominated by the results of an algorithm. The point is constructed with a vector of the objectives. Let $x \in \mathbb{R}^d$ be an arbitrary point in the objective space and $\eta=\{s_j \in \mathbb{R}^d, j = 1,2,\dots,M\}$ a solution set made of non-dominated elements. The attainment function is derived from equation (27). This function describes the probability that at least one solution weakly dominates x on a single run of the algorithm α. Since algorithms are stochastic and it is impossible to express the function in a closed form, the attainment function is empirically approximated (hence the term empirical attainment function) by the outcomes of running the algorithms repeatedly as defined by equation (28). In this equation, η_i $(i = 1,2,\dots,r)$ represents the Pareto set obtained in ith repetition of the algorithm α, and r indicates the number of repetitions for this algorithm. $I(\eta_i \trianglelefteq x) = 1$ when at least one solution in η_i weakly dominates x. To achieve the comparison of the two algorithms α and β, a differential function (*Diff-EAF*) between two EAFs is calculated using equation (29), which was proposed by López-Ibáñez et al. (2006b). This function expresses the probability of each solution x in the objective space being dominated by algorithm α but not by β.

Fig. 9 depicts the *EAFs* of our proposed RIPG and the MOSA. The *Diff-EAF* between the RIPG and MOSA is also given. These values are calculated after 100 runs of each algorithm over the instance with 89 tasks and 8 workers under *NI*×*20* millisecond termination criterion. In Fig. 9(a-b), the zones with intense colors indicate a high probability of being dominated while zones with light colors indicate unlikely dominance. In Fig. 9(c), the colors red and blue represent the positive and negative values of $Diff\text{-}EAF_{(\alpha,\beta)}$. From Fig. 9, it is clear that the proposed RIPG outperforms MOSA around the total ergonomic risk and cycle time spaces.

$$AF_\alpha(x) = P(\exists s_j \in \eta: s_j \trianglelefteq x) \tag{27}$$

$$EAF_\alpha(x) = \frac{1}{r}\sum_{i=1}^{r} I(\eta_i \trianglelefteq x) \tag{28}$$

$$Diff\_EAF_{(\alpha,\beta)}(x) = \frac{1}{r}\sum_{i=1}^{r}\left[I(\eta_i^\alpha \trianglelefteq x) - I\left(\eta_i^\beta \trianglelefteq x\right)\right] \tag{29}$$

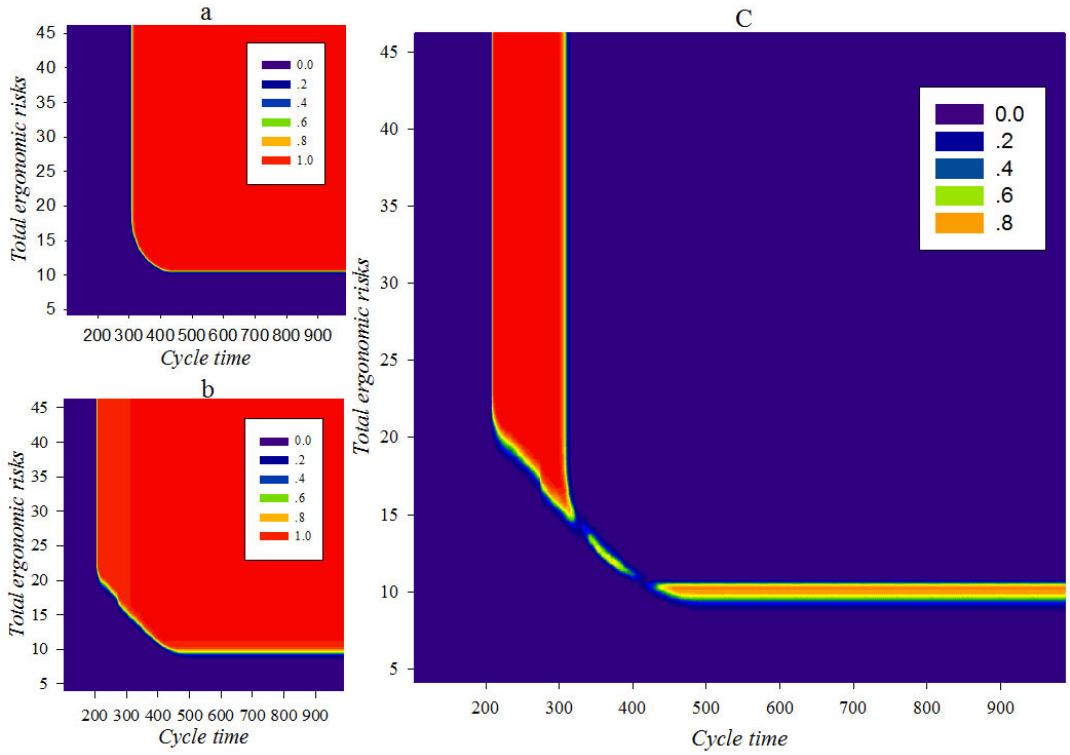


Fig. 9. Empirical Attainment Functions for MOSA (a) and RIPG (b), and the Differential Attainment Function between RIPG and MOSA (c).

6. Conclusions and future work

In this paper, we have studied the U-shaped worker assignment assembly line balancing problem where the assignment and ergonomic risks to workers have a great impact on assembly line performance and cannot be ignored. If we only optimize cycle times or productivity, the ergonomic risk values are high, as observed from our results. This demonstrates the need to jointly consider the objectives of cycle time and total ergonomic risk, and it constitutes the major contribution of this work. To achieve the optimization of both objectives, a new U-shaped assembly worker assignment and balancing problem considering ergonomic risks is formulated, where an OCRA method is applied to calculate the ergonomic risk value. A multi-objective Restarted Iterated Pareto Greedy (RIPG) algorithm is proposed to tackle this problem. In this algorithm, three improvements strategies are developed to enhance its performance: a problem-specific and heuristic-based initialization to find a good solution, two precedence-based greedy and local search phases to exploit the space around the current solution and a restart mechanism to help the algorithm escape from local optima. Finally, a full factorial design of experiments and the multifactor Analysis of Variance (ANOVA) is applied to verify the parameters of the proposed RIPG, and also to gauge the effectiveness and relevance of each improvement strategy in the RIPG.

Comprehensive statistical and computational experiments demonstrate that the proposed RIPG algorithm can achieve very good Pareto front approximations for a complex and relevant industrial problem when compared to nine other calibrated methods from the literature. We have employed two Pareto-compliant performance indicators, hypervolume ratio and unary epsilon indicators, to evaluate the Pareto front sets obtained by these algorithms. Differential Empirical Attainment Functions have also been used to infer the information about the spatial behavior of the tested

algorithms. All these experimental results suggest that the proposed RIPG outperforms all nine other algorithms.

Based on this work, future research will consider the collaboration between humans and robots to further enhance flexibility and productivity. Additionally, the NIOSH method is also fit for calculating the ergonomic risk values in assembly lines and hence future research can further utilize this method to estimate the ergonomic risks of manual handling. And then, the NIOSH method is compared with the OCRA method to judge which one is more suitable for assembly lines.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments and constructive suggestions. This work is supported by National Natural Science Foundation of China (No. 51875421, No. 51875420). Rubén Ruiz is partly supported by the Spanish Ministry of Science, Innovation, and Universities, under the project "OPTEP-Port Terminal Operations Optimization" (No.\ RTI2018-094940-B-I00) financed with FEDER funds.

References

Akyol, S.D., Baykasoğlu, A., 2016. ErgoALWABP: A multiple-rule based constructive randomized search algorithm for solving assembly line worker assignment and balancing problem under ergonomic risk factors. Journal of Intelligent Manufacturing, 1-12.

Alavidoost, M.H., Babazadeh, H., Sayyari, S.T., 2016. An interactive fuzzy programming approach for bi-objective straight and U-shaped assembly line balancing problem. Applied Soft Computing 40, 221-235.

Alavidoost, M.H., Tarimoradi, M., Zarandi, M.H.F., 2015. Fuzzy adaptive genetic algorithm for multi-objective assembly line balancing problems. Applied Soft Computing 34, 655-677.

Aryanezhad, M., Kheirkhah, A., Deljoo, V., Mirzapour Al-e-hashem, S., 2009. Designing safe job rotation schedules based upon workers' skills. The International Journal of Advanced Manufacturing Technology 41(1-2), 193-199.

Avikal, S., Jain, R., Mishra, P.K., Yadav, H.C., 2013. A heuristic approach for U-shaped assembly line balancing to improve labor productivity. Computers & Industrial Engineering 64(4), 895-901.

Aydoğan, E.K., Delice, Y., Özcan, U., Gencer, C., Bali, Ö., 2016. Balancing stochastic U-lines using particle swarm optimization. Journal of Intelligent Manufacturing, 1-15.

Battini, D., Calzavara, M., Otto, A., Sgarbossa, F., 2016. The integrated assembly line balancing and parts feeding problem with ergonomics considerations. IFAC-PapersOnLine 49(12), 191-196.

Battini, D., Faccio, M., Persona, A., Sgarbossa, F., 2011. New methodological framework to improve productivity and ergonomics in assembly system design. International Journal of Industrial Ergonomics 41(1), 30-42.

Bautista, J., Batalla-Garcia, C., Alfaro-Pozo, R., 2016. Models for assembly line balancing by temporal, spatial and ergonomic risk attributes. European Journal of Operational Research 251(3), 814-829.

Baykasoglu, A., 2006. Multi-rule multi-objective simulated annealing algorithm for straight and U type assembly line balancing problems. Journal of Intelligent Manufacturing 17(2), 217-232.

Baykasoğlu, A., Özbakır, L., 2006. Stochastic U-line balancing using genetic algorithms. The International Journal of Advanced Manufacturing Technology 32(1-2), 139-147.

Blum, C., Miralles, C., 2011. On solving the assembly line worker assignment and balancing problem

via beam search. Computers & Operations Research 38(1), 328-339.
Borba, L., Ritt, M., 2014. A heuristic and a branch-and-bound algorithm for the Assembly Line Worker Assignment and Balancing Problem. Computers & Operations Research 45, 87-96.
Bortolini, M., Faccio, M., Gamberi, M., Pilati, F., 2017. Multi-objective assembly line balancing considering component picking and ergonomic risk. Computers & Industrial Engineering 112, 348-367.
Botti, L., Mora, C., Regattieri, A., 2017. Integrating ergonomics and lean manufacturing principles in a hybrid assembly line. Computers & Industrial Engineering 111, 481-491.
Bukchin, Y., Raviv, T., 2018. Constraint programming for solving various assembly line balancing problems. Omega-Int J Manage S 78, 57-68.
Chaves, A.A., Lorena, L.A., Miralles, C., 2009. Hybrid Metaheuristic for the Assembly Line Worker Assignment and Balancing Problem, Hybrid Metaheuristics, International Workshop, Hm 2009, Udine, Italy, October 16-17, 2009. Proceedings. pp. 1-14.
Chaves, A.A., Miralles, C., Lorena, L.A.N., 2007. Clustering search approach for the assembly line worker assignment and balancing problem, Proceedings of the 37th international conference on computers and industrial engineering, Alexandria, Egypt. pp. 1469-1478.
Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182-197.
Ding, J.Y., Song, S.J., Gupta, J.N.D., Zhang, R., Chiong, R., Wu, C., 2015. An improved iterated greedy algorithm with a Tabu-based reconstruction strategy for the no-wait flowshop scheduling problem. Applied Soft Computing 30, 604-613.
Fattahi, A., Elaoud, S., Sadeqi Azer, E., Turkay, M., 2013. A novel integer programming formulation with logic cuts for the U-shaped assembly line balancing problem. International Journal of Production Research 52(5), 1318-1333.
Finco, S., Battini, D., Delorme, X., Persona, A., Sgarbossa, F., 2019. Workers' rest allowance and smoothing of the workload in assembly lines. International Journal of Production Research, 1-16.
Grunert da Fonseca, V., Fonseca, C.M., Hall, A.O., 2001. Inferential Performance Assessment of Stochastic Optimisers and the Attainment Function. 1993, 213-225.
Hatami, S., Ruiz, R., Andres-Romano, C., 2015. Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times. International Journal of Production Economics 169, 76-88.
Hazır, Ö., Dolgui, A., 2015. A decomposition based solution algorithm for U-type assembly line balancing with interval data. Computers & Operations Research 59, 126-131.
Hignett, S., McAtamney, L., 2000. Rapid entire body assessment (REBA). Applied ergonomics 31(2), 201-205.
Hoffmann, T., 1990. Assembly line balancing: a set of challenging problems. International Journal of Production Research 28(10), 1807-1815.
Karhu, O., Kansi, P., Kuorinka, I., 1977. Correcting working postures in industry: A practical method for analysis. Applied ergonomics 8(4), 199-201.
Knowles, J.D., Thiele, L., Zitzler, E., 2006. A tutorial on the performance assessment of stochastic multiobjective optimizers. TIK-Report 214.
Li, M., Tang, Q., Zheng, Q., Xia, X., Floudas, C.A., 2017. Rules–based Heuristic Approach for the U-Shaped Assembly Line Balancing Problem. Applied Mathematical Modelling 48, 423-439.
Li, Z.X., Tang, Q.H., Zhang, L.P., 2017. Two-sided assembly line balancing problem of type I: Improvements, a simple algorithm and a comprehensive study. Computers & Operations Research 79,

78-93.
Liles, D.H., Deivanayagam, S., Ayoub, M.M., Mahajan, P., 1984. A job severity index for the evaluation and control of lifting injury. Hum Factors 26(6), 683-693.
López-Ibáñez, M., Paquete, L., Stützle, T., 2006a. Hybrid Population-Based Algorithms for the Bi-Objective Quadratic Assignment Problem. Journal of Mathematical Modelling & Algorithms 5(1), 111-137.
López-Ibáñez, M., Paquete, L., Stützle, T., 2006b. Hybrid Population-Based Algorithms for the Bi-Objective Quadratic Assignment Problem. Journal of Mathematical Modelling and Algorithms 5(1), 111-137.
McAtamney, L., Nigel Corlett, E., 1993. RULA: a survey method for the investigation of work-related upper limb disorders. Applied ergonomics 24(2), 91-99.
Miltenburg, G.J., Wijingaard, J., 1994. The U-Line Line Balancing Problem. Management Science 40(10), 1378-1388.
Minella, G., Ruiz, R., Ciavotta, M., 2011. Restarted Iterated Pareto Greedy algorithm for multi-objective flowshop scheduling problems. Computers & Operations Research 38(11), 1521-1533.
Miralles, C., García-Sabater, J.P., Andrés, C., Cardós, M., 2008. Branch and bound procedures for solving the Assembly Line Worker Assignment and Balancing Problem: Application to Sheltered Work centres for Disabled. Discrete Applied Mathematics 156(3), 352-367.
Mutlu, O., Polat, O., Supciller, A.A., 2013. An iterative genetic algorithm for the assembly line worker assignment and balancing problem of type-II. Computers & Operations Research 40(1), 418-426.
Nourmohammadi, A., Zandieh, M., Tavakkoli-Moghaddam, R., 2013. An imperialist competitive algorithm for multi-objective U-type assembly line design. Journal of Computational Science 4(5), 393-400.
Occhipinti, E., 1998. OCRA: a concise index for the assessment of exposure to repetitive movements of the upper limbs. Ergonomics 41(9), 1290-1311.
Oksuz, M.K., Buyukozkan, K., Satoglu, S.I., 2017. U-shaped assembly line worker assignment and balancing problem: A mathematical model and two meta-heuristics. Computers & Industrial Engineering 112, 246-263.
Otto, A., Battaia, O., 2017. Reducing physical ergonomic risks at assembly lines by line balancing and job rotation: A survey. Computers & Industrial Engineering 111, 467-480.
Otto, A., Scholl, A., 2011. Incorporating ergonomic risks into assembly line balancing. European Journal of Operational Research 212(2), 277-286.
Pan, Q., Wang, L., Zhao, B.-H., 2007. An improved iterated greedy algorithm for the no-wait flow shop scheduling problem with makespan criterion. The International Journal of Advanced Manufacturing Technology 38(7-8), 778-786.
Pan, Q.K., Ruiz, R., 2014. An effective iterated greedy algorithm for the mixed no-idle permutation flowshop scheduling problem. Omega-Int J Manage S 44(2), 41-50.
Pereira, J., 2018. The Robust (minmax regret) assembly line worker assignment and balancing problem. Computers & Operations Research 93, 27-40.
Rabbani, M., Kazemi, S.M., Manavizadeh, N., 2012. Mixed model U-line balancing type-1 problem: A new approach. Journal of Manufacturing Systems 31(2), 131-138.
Rabbani, M., Mousavi, Z., Farrokhiasl, H., 2016. Multi-objective metaheuristics for solving a type II robotic mixed-model assembly line balancing problem. Journal of Industrial & Production Engineering 33(7), 1-13.

Ramezanian, R., Ezzatpanah, A., 2015. Modeling and solving multi-objective mixed-model assembly line balancing and worker assignment problem. Computers & Industrial Engineering 87, 74-80.
Ruiz, R., Stutzle, T., 2007. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European Journal of Operational Research 177(3), 2033-2049.
Sahin, M., Kellegoz, T., 2017. An efficient grouping genetic algorithm for U-shaped assembly line balancing problems with maximizing production rate. Memetic Computing 9(3), 213-229.
Saif, U., Guan, Z., Zhang, L., Zhang, F., Wang, B., Mirza, J., 2017. Multi-objective artificial bee colony algorithm for order oriented simultaneous sequencing and balancing of multi-mixed model assembly line. Journal of Intelligent Manufacturing, 1-26.
Schaub, K., Caragnano, G., Britzke, B., Bruder, R., 2013. The European assembly worksheet. Theoretical Issues in Ergonomics Science 14(6), 616-639.
Scholl, A., 1995. Balancing and sequencing of assembly lines. Publications of Darmstadt Technical University Institute for Business Studies.
Scholl, A., 1999. Balancing and Sequencing of Assembly Lines. Physica-Verlag Heidelberg, Germany.
Shin, W., Park, M., 2019. Ergonomic interventions for prevention of work-related musculoskeletal disorders in a small manufacturing assembly line. International journal of occupational safety and ergonomics : JOSE 25(1), 110-122.
Sungur, B., Yavuz, Y., 2015. Assembly line balancing with hierarchical worker assignment. Journal of Manufacturing Systems 37, 290-298.
Talbot, F.B., Patterson, J.H., Gehrlein, W.V., 1986. A Comparative Evaluation of Heuristic Line Balancing Techniques. Management Science 32(4), 430-454.
Tan, K.C., Yang, Y.J., Goh, C.K., 2006. A distributed cooperative coevolutionary algorithm for multiobjective optimization. Ieee Transactions on Evolutionary Computation 10(5), 527-549.
Tasgetiren, M.F., Kizilay, D., Pan, Q.K., Suganthan, P.N., 2017. Iterated greedy algorithms for the blocking flowshop scheduling problem with makespan criterion. Computers & Operations Research 77, 111-126.
Tiacci, L., Mimmi, M., 2018. Integrating ergonomic risks evaluation through OCRA index and balancing/sequencing decisions for mixed model stochastic asynchronous assembly lines. Omega 78, 112-138.
Vila, M., Pereira, J., 2014. A branch-and-bound algorithm for assembly line worker assignment and balancing problems. Computers & Operations Research 44, 105-114.
Waters, T.R., Putz-Anderson, V., Garg, A., Fine, L.J., 1993. Revised NIOSH equation for the design and evaluation of manual lifting tasks. Ergonomics 36(7), 749-776.
Yorke, J., 2017. Henry Ford – Master of flow.
Zacharia, P.T., Nearchou, A.C., 2016. A population-based algorithm for the bi-objective assembly line worker assignment and balancing problem. Engineering Applications of Artificial Intelligence 49, 1-9.
Zhang, Z., Tang, Q., Han, D., Li, Z., 2018. Enhanced migrating birds optimization algorithm for U-shaped assembly line balancing problems with workers assignment. Neural Computing and Applications.