Central European Journal of Operations Research
https://doi.org/10.1007/s10100-023-00885-x

Comparison of MILP and CP models for balancing partially automated assembly lines

Imre Dimény[1] · Tamás Koltai[2]

Accepted: 11 September 2023

Abstract

The objective of Assembly Line Balancing (ALB) is to find the proper assignment of tasks to workstations, taking into consideration various types of constraints and defined management goals. Early research in the field focused on solving the Simple Assembly Line Balancing problem, a basic simplified version of the general problem. As the production environment became more complex, several new ALB problem types appeared, and almost all ALB problems are NP-hard, meaning that finding a solution requires a lot of time, resources, and computational power. Methods with custom-made algorithms and generic approaches have been developed for solving these problems. While custom-made algorithms are generally more efficient, generic approaches can be more easily extended to cover other variations of the problem. Over the past few decades, automation has played an increasingly important role in various operations, although complete automation is often not possible. As a result, there is a growing need for partially automated assembly line balancing models. In these circumstances, the flexibility of a generic approach is essential. This paper compares two generic approaches: mixed integer linear programming (MILP) and constraint programming (CP), for two types of partially automated assembly line balancing problems. While CP is relatively slower in solving the simpler allocation problems, it is more efficient than MILP when an increased number of constraints is applied to the ALB and an allocation and scheduling problem needs to be solved.

Keywords Assembly line balancing · Mixed integer linear programing · Constraint programming · Human–robot collaboration

✉ Imre Dimény
dimeny.imre@gtk.bme.hu

Tamás Koltai
koltai.tamas@gtk.bme.hu

[1] Quantitative Social and Management Sciences Research Centre, Budapest University of Technology and Economics, Budapest, Hungary

[2] Department of Management and Business Economics, Budapest University of Technology and Economics, Budapest, Hungary

Published online: 26 September 2023

1 Introduction

Assembly lines are an integral part of mass production. Proper task assignment to workstations can help reduce manufacturing costs, increase production rates, and improve worker's workload balance. With the recent advancements in robotics, the use of industrial robots in manufacturing has continued to grow. The automation of assembly lines aims at achieving further cost reduction, productivity increase and improves the safety level of workers (Villani et al. 2018).

Despite the advantages of automation, creating a completely automated line is not always possible. Many assembly lines are too complex for robots to fully replace the flexibility of human workers, or the automation of certain tasks may not be economically feasible. It is also often the case that robots perform activities at a slower pace (Weckenborg et al. 2020).

Automated production lines and manual assembly lines represent two distinct paradigms within the manufacturing domain. Automated production lines rely on robotics, machinery, and computer-controlled processes to carry out tasks with high precision and consistency. On the other hand, manual assembly lines involve workers performing various tasks, introducing an element of variability inherent in human actions. Finding the best resource configuration and the proper level of automation frequently leads to the establishment of hybrid assembly lines where significant increase in productivity can be achieved while also improving working conditions. (Michalos et al. 2014; Tsarouchi et al. 2017).

Operator 4.0 envisions a symbiotic relationship between workers and robots (Romero et al. 2020), where each brings their strengths to the table. Collaborative assembly line balancing enables the capabilities of both workforce types (Cohen et al. 2019).

The difficulty of solving the resulting assembly line balancing problems depends on several factors, including the size of the production line, the available resources, and other specific requirements of the production process. In general, finding the optimal solution to an assembly line balancing problem can be a computationally intensive task, and it may require significant time and effort to obtain a satisfactory solution.

Assembly line balancing problems can be solved with exact or heuristic methods. The main difference between exact and heuristic assembly balancing models consist in the guarantee of finding the optimal solution. Exact models provide such a guarantee, while heuristic models do not. The exact method can be further split to generic methods and dedicated exact methods. The choice of method depends on the specific problem requirements and the available resources.

Balancing assembly lines with a hybrid setup requires models and approaches that take into consideration the nature and level of human–robot collaboration applied in the line. With the evolution of industrial robots an increasing number of different formulations already exists (Nourmohammadi et al. 2022) and new formulations of the line balancing models are expected. This means that a generic solution could be adapted more easily to the change in requirements than would be the case with a dedicated exact method.

The paper presents and compares mixed integer linear programming (MILP) and constraint programming (CP) formulations of three types of assembly line balancing problems for human only and partially automated lines, each building upon the previous one in terms of generality. In the first type only human workers are performing tasks. In the second type humans and robots are only used at separate stations and an allocation problem is solved. Finally, in the third type, at most one robot and one worker can be used at a station. In this case workers and robots are executing some tasks in parallel with high interdependency resulting from possible precedence relations between tasks. Considering the execution order of tasks assigned to the same station but to different resources converts the problem into an allocation and scheduling problem.

The remainder of this paper is organised as follows. In Sect. 2, an overview of the literature of the related assembly line balancing problems and possible approaches to solve is given. Section 3 outlines the mathematical formulations of the proposed models. In Sect. 4, the performance of the CP and MILP formulations of the proposed models are compared using benchmark datasets. Finally, the conclusions of the presented research are summarised in Sect. 5.

2 Literature review

An assembly line is a system of workstations that are used to complete a set of tasks on a product unit. The product unit moves through the line and is finished once it has been through all the workstations. Assembly line balancing problems (ALBPs) occur when designing or redesigning an assembly line, and they involve finding the best way to assign tasks to workstations to meet production goals. A well-planned allocation of tasks can improve profit, increase the production rate, and improve workload efficiency, while also reducing costs, cycle times, idle times, and the number of workers needed.

The Simple Assembly Line balancing problem (SALBP), is a basic simplified version of the general problem that assumes indivisible tasks, a fixed maximum cycle time, deterministic operation times, production of just one homogeneous product, no assignment restrictions besides the precedence constraints and a single line with equally equipped stations (Becker and Scholl 2006).

The assumptions of the simple assembly line balancing problem simplify the problem and make it easier to solve, but they may not reflect the real-world complexity of a manufacturing or production environment. In practice, many of these assumptions are relaxed and the problem becomes more complex, requiring more sophisticated methods for solving it. Sivasankaran and Shahabudeen (2014) offers a broad overview of the different types of ALBPs, categorized based on their assumptions and solution methods.

Modern assembly lines use industrial robots to execute certain tasks on the line. In cases where a full automation is not possible both workers and robots execute tasks on the line, leading to a necessary collaboration between the humans and robots. Human–robot collaboration arises when robots and workers work together in a shared physical workspace, workers and robots work towards a shared goal in shared space or workers and robots work simultaneously on a shared object (Arents et al. 2021).

In the case of a partially automated line the type of resource used to perform certain tasks also needs to be decided. Finding the proper assignment is enough on its own only if tasks can be executed in an arbitrary order at the station without affecting the cycle time or feasibility. However, the execution order of tasks at the station may influence the station time in many cases, and an incorrect execution order could render the solution infeasible. When the execution order of tasks at the station is also taken into consideration, the model leads to there being a joint assignment and scheduling problem.

Many papers have been published relating to assembly lines that use multiple resource types with various additional constraints to which partially automated assembly line balancing problems can be related. Parallelisation and resource assignment are two important characteristics of ALB model classification (Boysen et al. 2007) that most obviously relate most to partially automated lines.

Two-sided assembly lines (TALBPs) are one example of possible parallelisation and consist of a series of mated-stations, each featuring two facing sides. This type of parallelization allows two different tasks to be performed simultaneously on the same piece if the two tasks do not interfere with each other. Approaches towards solving the two sided assembly line problems include heuristic-based assignment procedures (Lee et al. 2001), branch-and-bound algorithm (Wu et. al. 2008), tabu search algorithm (Özcan and Toklu 2009), genetic algorithm (Purnomo et al. 2013) and constraint programming (Kizilay and Cil 2020).

A significant amount of research has been conducted on the resource assignment aspect of assembly lines, particularly regarding the types of resources utilised in these systems. This literature covers a wide range of topics, including optimising resource allocation and using advanced technologies such as robotics and automation.

The Multi-manned Assembly Line Balancing Problem (MALBP) typically occurs in industries producing high volume of large size products (Giglio et al 2017) and allows the assignment of more than one operator with identical skills and equal processing times to each workstation. Miralles et al. (2007) considers worker dependent processing times in their model and formulate the assembly line worker assignment and balancing problem (ALWABP). Yilmaz and Yilmaz (2020) more recently formulated a mathematical model for multi-manned assembly lines with assignment restrictions and proposed a tabu search algorithm to solve the resulting problem. Roshani and Giglio (2015) present a simulated annealing algorithm for solving the MALBP.

Pinto et al. (1983) extend the simple assembly line balancing problem to include alternative processing options. Tasks are still executed by workers, but they do this using non-identical equipment. The dual problem of assigning equipment to workstations and tasks to workstations gave rise to the equipment selection problem. Graves and Withney (1979) presented a linear programming model to solve the problem for a single product. Later, Bukchin and Tzur (2000) developed an optimal method and a heuristic algorithm that can be used when several equipment alternatives are available.

The assembly line that uses robots in its value-added operations is referred as the Robotic Assembly Line Balancing problem (RALBP) and was first formulated by Rubinovitz et al. (1993). A heuristic algorithm is proposed for selecting the robots to perform the tasks on workstations with an objective of minimising the number of workstations for a given cycle time. In this case the deterministic execution time of tasks depends on the robots assigned to the station.

In advanced manufacturing systems, where humans and collaborative robots share the same workplace and can simultaneously perform tasks the human-collaborative assembly line balancing and scheduling problem (HRCALBSP) arises. A few recent papers discuss the case when robots and workers can perform tasks on the same station separately or jointly.

Weckenborg et al. (2020) developed a genetic algorithm for solving the HRCALBSP. In their solution, collaborative tasks can be performed by workers and robots jointly and in parallel. A mixed-integer program has been formulated to optimise the assignment of collaborative robots to stations and the distribution of workload between human workers and robots in an assembly line. Koltai et al. (2021) presented models for analysing the task assignment and cycle times of the assembly lines when robots are added to the line, taking into consideration the possible interferences between the workers and robots. Dimény and Koltai (2022) proposed a MILP model to evaluate worker's workload in partially automated assembly lines depending on the number of available robots. Dalle Mura and Dini (2019) proposes a genetic algorithm for ALB problems in case of human–robot collaborative work. The aim of the algorithm is the minimization of the assembly line cost and minimization of the number of skilled workers on the line. A mathematical model and bee algorithm is presented by Çil et al. (2020) for solving the mixed-model assembly line balancing problem whith physical human–robot collaboration. Stecke and Mokhtarzadeh (2022) developed a mixed-integer linear programming model, a constraint programming model, and a Benders decomposition algorithm to analyse advantages of collaborative robots in assembly lines.

A mixed-integer programming model for balancing assembly lines under consideration of ergonomic and economic objectives and the availability of novel technologies like exoskeletons is developed and presented by Weckenborg et al. (2022). Battaïa and Dolgui (2022) offers a comprehensive review on new line balancing trends and formulations.

The ALBP in general is a complex and well-known combinatorial optimisation problem. The ALB problem was first formulated as a linear programming (LP) model by Salveson (1955). Bukchin and Raviv (2018) showed that constraint programming (CP) formulation outperformes mixed-integer linear programming (MILP) formulation for medium to large problem instances when solving the SALB problem. The assembly line balancing related optimisation problems and possible strategies have been extensively discussed in the literature (see for example, Scholl 1993; Thomopoulos 2014). Besides a taxonomy of line balancing problems, Battaïa and Dolgui (2013) present solution approaches used to solve ALBPs.

The solution approaches of ALBPs can be classified into two main groups: exact methods and approximate methods (Topaloglu et al. 2012). The key difference between exact and approximate methods is the proven optimality of the exact methods. Approximate metods like heuristic and meta-heuristic algorithms can provide acceptable good solutions within reasonable computational time.

The exact methods result in a proven optimal solution and can be found either using a problem specific exact method or with a generic solution using standard solvers. In the first case, the goal is to design a problem specific algorithm that makes use of the special characteristics of the assembly process. Generic solutions take advantage of existing general solvers. With generic solutions like CP or MILP the goal is to find the most appropriate model for the problem and adjust to the requirements by modifying the objective function or adding constraints.

Dedicated exact methods in many cases are more efficient and can cope with larger problems, generic solutions have the advantage of easier implementation and faster adaptation to changes related to the constraints to be applied (Battaïa and Dolgui 2013).

In this paper two generic exact methods, mixed integer linear programming (MILP) and constraint programming (CP) to minimise the number of workers in three types of assembly line balancing problems are compared. In the first model only workers are available. In the second model, each station is either completely human operated or automated; thus, an allocation problem (SALB with resource selection) is solved. In the third model at most one worker and one robot executes tasks assigned to stations. In this case, the execution order of tasks within the station must also be considered, making the problem an allocation and scheduling problem (MALBP-SW model). In all three cases the objective is to minimise the number of workers required.

3 Mathematical descriptions of the problems

In this section the mathematical formulation of the three problems (SALB, SALB with resource selection and MALBP-SW) using two approaches (MILP and CP) is presented.

The three models have the following common attributes:

- All activities can be executed by human workers,
- The processing of tasks with robots is limited to a predefined set of tasks,
- Task times are deterministic for each type of resource separately,
- The precedence relations of tasks are defined,
- The maximum allowed cycle time is defined.

Notations used in the paper are summarised in Table 1.

The mathematical model in Table 2 contains the MILP formulation of the three models. Objective function (1) and constraints (2), (3), (4), (5) and (6) form the model for minimising the number of workers when only workers can be assigned to station (MILP1).

The generated model is equal to the SALBP1 model. As of constraint (2), (3) and (4) all tasks are allocated to a station and workforce type in such way that the total processing time on any of the station does not exceed the predefined T_c cycle

Table 1 Notation used in the paper

Indices	
i, k	Index of tasks $(1, \dots, I)$,
j	Index of workstations $(1, \dots, J)$,
w	Index of workforce types ({H (human worker), R (robot)}),
l	Index of final tasks,
Parameters:	
I	Number of tasks,
J	Maximum number of workstations,
N^*	Minimum number of stations (the result of the station number minimisation model)
t_{iw}	Time necessary to perform task i using workforce type w (task time),
T_c	Cycle time of the assembly line,
Sets:	
W	Set of workforce types containing two elements: worker and robot,$\{H, R\}$,
L	Set of final tasks, $i \in L$, if task i does not precede any other task,
P_i	Set of indices of those tasks which must be finished before task i is started
NA_w	Set of tasks for which resource type w has no ability,
WA_i	Set of workforce types which have ability to execute task i;$\{w \mid i \notin NA_w\}$
Decision variables:	
x_{ijw}	0–1 Decision variable; if $x_{ijw} = 1$ then task i is assigned to station j using workforce type w, otherwise $x_{ijw} = 0$,
xe_i	$xe_i \in \{1, .., J\}$; if $xe_i =$ j then task i is assigned to station j,
y_{ik}	0–1 Decision variable; if $y_{ik} = 1$ then task i and k are assigned to the same station and i is executed before k,
u_{jw}	0–1 Decision variable; if $u_{jw} = 1$ then workforce type w is used on station j,
ue_j	$ue_j \in \{H, R\}$; if $ue_j = w$ then workforce type w is assigned to station j,
s_i	Continuous decision variable: start time of task i on the respective station,
e_i	Continuous decision variable: end time of task i on the respective station,
a_i	Main activity related to each task, with decision variables $a_i.present$, $a_i.start$, $a_i.end$,$a_i.length$
a_{ijw}	Activity related to executing task i on station j using workforce type w, with decision variables $a_{ijw}.present$, $a_{ijw}.start$, $a_{ijw}.end$, $a_{ijw}.length$. Exactly one activity will be present for all (j, w) combinations of station and workforce type
r_{jw}	Sequential resource needed for activities
N	Objective function variable for the number of workers used

time while precedence relations among tasks are satisfied. As of constraint (5) tasks can only be allocated to workers. Constraint (6) defines the minimum number of workers required as the maximum station value to which task is allocated.

Objective function (1) and constraints (2), (3), (4), (7), (8), (9), (10), and (11) form the model for minimising the number of workers when each station is either completely human operated or automated (MILP2). As of constraint (7) each task must be allocated to a workforce type with ability to execute the task. As of constraint (8) exactly one type of workforce is allocated to each station.

Table 2 MILP formulations of the three models

		MILP1	MILP2	MILP3	
$Min(N)$		x	x	x	(1)
$\sum_{j,w} x_{ijw} = 1$	$\forall i$	x	x	x	(2)
$\sum_{j,w} j\left(x_{ijw} - x_{kjw}\right) \geq 0$	$\forall (i,k) \mid k \in P_i$	x	x	x	(3)
$\sum_i x_{ijw} t_{iw} \leq T_c$	$\forall (j,w)$	x	x		(4)
$x_{ijw} = 0$	$\forall (i,j,w) \mid w = R$	x			(5)
$N \geq \sum_l j * x_{ljw}$	$\forall (j,w)$	x			(6)
$x_{ijw} = 0$	$\forall (i,j,w) \mid i \in NA_w$		x		(7)
$\sum_w u_{jw} = 1$	$\forall j$		x		(8)
$\sum_i x_{ijw} \leq I \bullet u_{jw}$	$\forall (i,w)$		x	x	(9)
$\sum_i x_{ijw} \geq u_{jw}$	$\forall (i,w)$		x	x	(10)
$N = \sum_j u_{jH}$			x	x	(11)
$\sum_w u_{jw} \geq 1$				x	(12)
$y_{ii} = 0$	$\forall i$			x	(13)
$y_{ik} + y_{ki} = 1$	$\forall (i,k) \mid k \notin P_i, i \notin P_k, i \neq k$			x	(14)
$y_{ik} \geq \sum_{j_1,w \mid j_1<j} x_{ij_1w} + x_{kjw} - 1$	$\forall (i,k,j) \mid k \notin P_i, i \notin P_k, i \neq k$			x	(15)
$e_i \geq 0; s_i \geq 0$	$\forall i$			x	(16)
$e_i = s_i + \sum_{j,w} x_{ijw} t_{iw}$	$\forall i$			x	(17)
$e_i \leq T_c$	$\forall i$			x	(18)
$s_i \geq e_k - T_c\left(2 - \sum_w \left(x_{ijw} + x_{kjw}\right)\right)$	$\forall (i,k,j) \mid k \in P_i$			x	(19)
$s_k \geq e_i - T_c\left(3 - y_{ik} - x_{ijw} - x_{kjw}\right)$	$\forall (i,k,j,w) \mid k \notin P_i, i \notin P_k, i \neq k$			x	(20)
$s_k \geq s_i - T_c\left(3 - y_{ik} - \sum_w x_{ijw} - \sum_w x_{kjw}\right)$	$\forall (i,k,j) \mid k \notin P_i, i \notin P_k, i \neq k$			x	(21)

Constraint (9) and (10) defines the relation between the allocation of tasks to stations and workforces to the allocation of workforces to stations. Tasks can only be allocated to a station and workforce type if the same workforce type is allocated to the station. If a task is allocated to a station and workforce type, the workforce type should also be allocated to the station. The objective value N is the equal to the number of stations with workers allocated to them as defined by constraint (11).

Objective function (1) and constraints (2), (3), (4), (9), (10), (11), (12), (13), (14), (15), (16), (17), (18), (19), (20) and (21) form the model for minimising the number of workers when at most one worker and one robot can be allocated to a station and task are executed at the stations either sequentially by the same workforce type or in parallel by a worker and a worker (MILP3).

As of constraint (12) at least one workforce type is allocated to a station. In this model an execution precedence variable y_{ik} is introduced for tasks with no precedence relation predefined. Constraints (13), (14) and (15) define the constraints related to these variables. No task can precede itself (13), for all pair of tasks with no predefined precedence an execution precedence exists (14), and the execution precedence is defined by the station number for tasks allocated to different stations and with no predefined precedence relation between them (15).

Constraints (16), (17) and (18) define key properties of the start and end time of each activity. All start and end times are positive values (16), end time is equal to the start time plus the processing time (17) and the predefined cycle time is greater or equal to the end time of any of the tasks (18).

Additional constraints are defined for pairs of tasks allocated to same station by constraints (19), (20) and (21). A task may begin only after all of its preceding tasks allocated to the same station have already been completed (19). Two constraints are defined for pairs of tasks allocated to the same station with no precedence relation between them. First, any task scheduled to take place in front of another task must be completed before the other can be started on the same station as of constraint (20). Secondly, any task scheduled in front of another task and allocated to a different workforce type must be started earlier than the other task as of constraint (21).

Table 3 contains the constraint programming formulation counterpart of the MILP1, MILP2, MILP3 models. Objective function (22) and constraints (23) and (24) form the CP1 model for minimising the number of workers when only workers are available on stations. This is the constraint programming formulation of the SALBP-1 problem. This formulation describes the problem as a bin packing problem where preceding tasks must be allocated to bins numbered lower or equal to the bin number of succeeding task. The bin packing constraints are defined by constraint (23). The precedence relation is defined by constraint (24).

Objective function (22) and constraint (24), (25), (26) and (27) form the CP2 constraint programming model for minimising the number of workers when workers and robots are also available, but only one workforce type can be assigned to a station. This is the CP formulation counterpart of the MILP2 model. In this formulation

Table 3 CP formulation of the three models

		CP1	CP2	CP3	
$Min(N)$		x	x	x	(22)
$Bin\,Packing\,(j;T_c;(i);xe_i;t_i,N)$		x			(23)
$xe_i \le xe_k$	$\forall(i,k)\vert k \in P_i$	x	x		(24)
$ue_{xe_i} \in WA_i$			x		(25)
$\sum_i \left(xe_i = j\right) \bullet t_{i,ue_j} \le T_c$	$\forall j$		x		(26)
$N = \sum_j \left(ue_j = H\right)$			x		(27)
$a_{ijw}.length = t_{i,w}$	$\forall(i,j,w)$			x	(28)
$\sum_{j,w} j\left(a_{ijw}.present - a_{kjw}.present\right) \ge 0$	$\forall(i,k)\vert k \in P_i$			x	(29)
$a_{ijH}.end \le T_c$	$\forall(i,j,w)$			x	(30)
$Alternate\left(a_i;(j,w);a_{ijw}\right)$	$\forall i$				(31)
$Sync\left(a_i;(j,w);a_{ijw}\right)$	$\forall i$			x	(32)
$End\,Before\,Begin(a_{ijH};a_{kjR})$	$\forall i,k,j\vert k \in P_i, k \notin NA_R$			x	(33)
$EndBefore\,Begin(a_{ijR};a_{kjH})$	$\forall i,k,j\vert k \in P_i, i \notin NA_R$			x	(34)
$Sequential\,Resource(r_{jw})$	$\forall(j,w)$			x	(35)
$Activity\,Resource\,Need(a_{ijw};r_{jw})$	$\forall(j,w)$			x	(36)
$N = \sum_j Exists(i\vert a_{ijH}.present)$				x	(37)

element variables are used to capture the decision related to the allocation of tasks to workstations and resources. As of constraint (25), tasks can only be allocated to a station that has workforce type allocated with ability to execute the task. Constraint (26) assures that the total processing time on any of the station does not exceed the predefined T_c cycle time, considering the workforce type dependent execution time. Constraint (27) defines the minimum number of workers required as the number of stations with workers assigned to it.

Objective function (22) and constraints (28), (29), (30), (31), (32), (33), (34), (35), (36) and (37) form the CP3 constraint programming model for minimising the number of workers when at most one worker and one robot is assigned to a station. This is the constraint programming counterpart of the MILP3 model. When both workers and robots can execute tasks on the same station the execution order of the tasks scheduled to the same station must be also considered. This turns the problem into an allocation and scheduling problem. The CP3 formulation uses activities with start, end, lengths, present variables and sequential resources related to each station and workforce type combination. A mandatory a_i and a set of optional a_{ijw} activity is defined for each task. The optional activities describe the potential allocations of tasks to stations and workforce types. Constraint (28) defines the length of the activities as the execution time required considering the workforce type used. Constraint (29) assures that the precedence relations defined are kept. Constraint (30) assures that the total processing time on any of the station does not exceed the predefined T_c cycle time and all activities are finished earlier than T_c. According to constraint (31), for each task i each activity a_i has exactly one manifestation as a_{ijw}. Constraint (32) requires that a_i activity start, end and lengths are defined by chosen a_{ijw} activity start, end and length. Constraint (33) and (34) dictates that activities with precedence relation among them and allocated to the same station but different resources must keep the defined execution order. According to constraint (35) activities allocated to the same station and workforce type can only be executed sequentially. Constraint (36) defines the resource need of activities a_{ijw}. Finally, constraint (37) defines the minimum number of workers required as the number of stations with workers assigned to it.

4 Comparison of the computational performances

The computational performance of the presented models was evaluated using benchmark datasets from Scholl (1993); each dataset is referred to by its name. Tasks in these problems are denoted by integer numbers. As these datasets represent scenarios where only one workforce type was available, the following assumptions were made:

Each task can be performed by workers. Robots can perform tasks numbered 1, 3, 4, 6, 7, 11, 19, 20, 22, 26, 27, 29, 32, 33, 35, 40, and 46 to 75 where such task exists.

The execution of tasks by a robot is considered slower than when executed by a worker. The task time of robotic operation is 150% of the task time of workers operation. Proper rounding is applied to get integer values for all task times.

The complexity of the datasets is described by the following indicators:

Shortest task duration (STD): the minimum value of all the task times.

Longest task duration (LTD): the maximum value of all the task times.

Order strength (OS): the percentage of the number of arcs in the transitive closure of the precedence graph related to the maximal number of arcs in an acyclic graph with I number of nodes.

Time variability ratio (TV): the ratio of the longest task duration to the shortest task duration.

Computational time was evaluated when using a laptop computer with 1.8 GHz Intel i7 processor and 16 GB of RAM. The models were build using AIMMS 4.90, MILP models were solved using CPLEX 22.1 solver, while CP models were solved using CP Optimizer 22.1.

Table 4 details the results and computation times related to the three models defined in Sect. 3. The bold face numbers in the table are the optimal values of the respective objective functions. During the computation, the optimal solution of case I was used as a warm start for case II and the optimal solution of case II was used as a warm start for the case III calculation. When the required computational time exceeded 10 min, the gap percentage after 10 min of computation and the solution (in parenthesis) when it differs from the optimal solution can be found in the corresponding “CPU time” row.

Analysing the results and computational time, the following characteristics can be observed:

Case I is equally rapidly solved for the datasets tested both using MILP and CP formulation.

In some cases, the Case II model is more difficult to solve for the CP formulation and in four instances (Arcus1, Kilbridge, Tonge, Mukherjee) the optimal value was not found during the 10 min run while the MILP formulation was able to quickly find solution for each dataset.

The Case III problem is much more complex and requires that a special scheduling problem be solved. The MILP formulation was not able to find an optimal solution within the 10-min run for datasets Arcus1, Lutz3, Kilbridge, Tonge, Mukherjee. The computation time required to solve the other datasets was higher than solving the problem with the CP formulation except for the Roszieg dataset which was solved easily by both formulation.

To sum up, the presented computational data indicate that MILP formulations are more suitable for more complex assembly line allocation problems. For solving assembly line allocation and scheduling problems, the CP formulation could be more efficient. Another advantage of using CP formulation is the easy definition and readability of the model, which makes modifications easier to implement when required.

5 Conclusions

In this paper, the possibility of using two generic approaches (mixed-integer linear programming and constraint programming) to find the exact optimum arrangement of partially automated assembly lines was investigated and compared with the basic

Table 4 Summary of the computational results

	Arcus 1	Gunther	Lutz 3	Buxey	Hahn	Roszieg	Kilbridge	Sawyer	Tonge	Mukherjee
Case 1										
I	83	35	89	29	53	25	45	30	70	94
STD	233	1	1	1	40	1	3	1	1	8
LTD	3691	40	74	25	1775	13	55	25	156	171
TV	15.8	40	74	25	44.4	13	18.3	25	156	21.38
OS	3.32	7.56	3.01	8.87	83.8	10.67	6.26	7.36	3.56	4.14
T_c	8412	41	150	27	2004	14	57	30	527	351
N	10	14	12	13	8	10	10	12	10	13
MILP1 CPU time (s)	1.75	0.61	2.55	0.2	0.05	0.05	0.17	0.28	1.27	1.78
CP1 CPU time (s)	0.28	0.25	1.13	0.05	0.16	0.03	0.09	0.06	0.14	0.38
N	8	12	10	12	6	9	10	10	9	11
Case 2										
MILP2 CPU time (s)	3.88	2.87	9.36	0.72	0.06	0.19	1.18	6.96	15.45	28.06
CP2 CPU time (s)	*25%*	4.37	23.53	0.84	0.17	0.13	*10%*	41.89	*33.3%*	*36.3%*
N	6	10	8	10	6	8	8	7	5	7
Case 3										
MILP3 CPU time (s)	*62.5%*	225.38	*20%*	50.84	80.59	2.66	*20%*	169.33	*50%*	*72%*
CP3 CPU time (s)	576.2	20.55	324.5	8.25	10.36	4.84	291	16.31	240.39	196.17

setup of having one workforce type. The two generic approaches were applied to three cases. In the basic case (case I) no automation was allowed, and the SALB1 problem was solved, so as to minimise the number of workers required. In case II, partial automation of certain tasks was possible, but stations were either completely automated or manual. In case III, at most one worker and one robot was assigned to each station and tasks within the station were executed either sequentially using the same workforce type or in parallel by a worker and a robot. The worker and the robot assigned to the same station can work on two different tasks at the same time. In case III, the execution order of activities at the station must also be taken into consideration, making the problem an allocation and scheduling problem. Besides the scheduling aspect, the fact that the complexity of the problem increases significantly must also be taken into consideration.

It can be noted that while the MILP models are built step by step, extending the formulation with additional variables and constraints, the CP formulation requires different modelling archetypes for the three cases (bin packing problem, allocation problem, scheduling problem).

As line configurations become more complex generic modelling approaches have the advantage of easier adaptability to the changing requirements. The present model formulation shows that both approaches can be used for formulating special cases of assembly line balancing problems. However, computational analysis shows that MILP formulations could better fit for allocation problems, while the CP formulation performs better when the scheduling aspect of the problem must also be considered.

Author contributions Imre Dimény: Investigation, Software, Formal analysis, Visualization, Writing—Original draft preparation. Tamás Koltai: Conceptualization, Investigation, Methodology, Supervision, Writing—Review & Editing.

Funding Open access funding provided by Budapest University of Technology and Economics.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

Arents J, Abolins V, Judvaitis J, et al (2021) Human–robot collaboration trends and safety aspects: A systematic review. J Sens Actuator Netw 10:48. https://doi.org/10.3390/jsan10030048

Battaïa O, Dolgui A (2013) A taxonomy of line balancing problems and their solution approaches. Int J Prod Econ 142:259–277. https://doi.org/10.1016/j.ijpe.2012.10.020

Battaïa O, Dolgui A (2022) Hybridizations in line balancing problems: a comprehensive review on new trends and formulations. Int J Prod Econ 250:108673. https://doi.org/10.1016/j.ijpe.2022.108673

Becker C, Scholl A (2006) A survey on problems and methods in generalized assembly line balancing. Eur J Oper Res 168:694–715. https://doi.org/10.1016/j.ejor.2004.07.023

Boysen N, Fliedner M, Scholl A (2007) A classification of assembly line balancing problems. Eur J Oper Res 183:674–693. https://doi.org/10.1016/j.ejor.2006.10.010

Bukchin J, Tzur M (2000). IIE Trans 32:585–598. https://doi.org/10.1023/a:1007646714909

Bukchin Y, Raviv T (2018) Constraint programming for solving various assembly line balancing problems. Omega 78:57–68. https://doi.org/10.1016/j.omega.2017.06.008

Çil ZA, Li Z, Mete S, Özceylan E (2020) Mathematical model and bee algorithms for mixed-model assembly line balancing problem with physical human–robot collaboration. Appl Soft Comput 93:106394. https://doi.org/10.1016/j.asoc.2020.106394

Cohen Y, Naseraldin H, Chaudhuri A, Pilati F (2019) Assembly systems in Industry 4.0 era: a road map to understand assembly 4.0. Int J Adv Manuf Technol 105:4037–4054. https://doi.org/10.1007/s00170-019-04203-1

Dalle Mura M, Dini G (2019) Designing assembly lines with humans and collaborative robots: a genetic approach. CIRP Ann Manuf Technol 68:1–4. https://doi.org/10.1016/j.cirp.2019.04.006

Dimény I, Koltai T (2022) Minimising workers' workload in partially automated assembly lines with human-robot collaboration. IFAC-PapersOnLine 55:1734–1739. https://doi.org/10.1016/j.ifacol.2022.09.648

Giglio D, Paolucci M, Roshani A, Tonelli F (2017) Multi-manned assembly line balancing problem with skilled workers: a new mathematical formulation. IFAC-PapersOnLine 50:1211–1216. https://doi.org/10.1016/j.ifacol.2017.08.344

Graves SC, Whitney DE (1979) A mathematical programming procedure for equipment selection and system evaluation in programmable assembly. In: Proceedings of the IEEE decision and control, pp 531–536

Kizilay D, Çil ZA (2020) Constraint programming approach for multi-objective two-sided assembly line balancing problem with multi-operator stations. Eng Optim 1–16. https://doi.org/10.1080/0305215x.2020.1786081

Koltai T, Dimény I, Gallina V, et al (2021) An analysis of task assignment and cycle times when robots are added to human-operated assembly lines, using mathematical programming models. Int J Prod Econ 242:108292. https://doi.org/10.1016/j.ijpe.2021.108292

Lee TO, Kim Y, Kim YK (2001) Two-sided assembly line balancing to maximize work relatedness and slackness. Comput Ind Eng 40:273–292. https://doi.org/10.1016/s0360-8352(01)00029-8

Michalos G, Makris S, Spiliotopoulos J et al (2014) ROBO-PARTNER: seamless human-robot cooperation for intelligent, flexible and safe operations in the assembly factories of the future. Procedia CIRP 23:71–76. https://doi.org/10.1016/j.procir.2014.10.079

Miralles C, García-Sabater JP, Andrés C, Cardos M (2007) Advantages of assembly lines in Sheltered Work Centres for Disabled: a case study. Int J Prod Econ 110:187–197. https://doi.org/10.1016/j.ijpe.2007.02.023

Nourmohammadi A, Fathi M, Ng AHC (2022) Balancing and scheduling assembly lines with human-robot collaboration tasks. Comput Oper Res 140:105674. https://doi.org/10.1016/j.cor.2021.105674

Özcan U, Toklu B (2009) A tabu search algorithm for two-sided assembly line balancing. Int J Adv Manuf Technol 43:822–829. https://doi.org/10.1007/s00170-008-1753-5

Pinto PA, Dannenbring DG, Khumawala BM (1983) Assembly line balancing with processing alternatives: an application. Manage Sci 29:817–830. https://doi.org/10.1287/mnsc.29.7.817

Purnomo HD, Wee H-M, Rau H (2013) Two-sided assembly lines balancing with assignment restrictions. Math Comput Model 57:189–199. https://doi.org/10.1016/j.mcm.2011.06.010

Romero D, Stahre J, Taisch M (2020) The Operator 4.0: towards socially sustainable factories of the future. Comput Ind Eng 139:106128. https://doi.org/10.1016/j.cie.2019.106128

Roshani A, Giglio D (2015) A simulated annealing approach for multi-manned assembly line balancing problem type II. IFAC-PapersOnLine 48:2299–2304. https://doi.org/10.1016/j.ifacol.2015.06.430

Rubinovitz J, Bukchin J, Lenz E (1993) RALB: a heuristic algorithm for design and balancing of robotic assembly lines. CIRP Ann Manuf Technol 42:497–500. https://doi.org/10.1016/s0007-8506(07)62494-9

Salveson ME (1955) The assembly line balancing problem. J Ind Eng 6:18–25

Scholl A (1993) Data of assembly line balancing problems. Schriften zur Quantitativen Betriebswirtschaftslehre 16/93, TU Darmstadt

Sivasankaran P, Shahabudeen P (2014) Literature review of assembly line balancing problems. Int J Adv Manuf Technol 73:1665–1694. https://doi.org/10.1007/s00170-014-5944-y

Stecke KE, Mokhtarzadeh M (2022) Balancing collaborative human–robot assembly lines to optimise cycle time and ergonomic risk. Int J Prod Res 60:25–47. https://doi.org/10.1080/00207543.2021.1989077

Thomopoulos NT (2014) Assembly line planning and control. Springer, Cham

Topaloglu S, Salum L, Supciller AA (2012) Rule-based modeling and constraint programming based solution of the assembly line balancing problem. Expert Syst Appl 39:3484–3493. https://doi.org/10.1016/j.eswa.2011.09.038

Tsarouchi P, Matthaiakis A-S, Makris S, Chryssolouris G (2017) On a human-robot collaboration in an assembly cell. Int J Comput Integr Manuf 30:580–589. https://doi.org/10.1080/0951192x.2016.1187297

Villani V, Pini F, Leali F, Secchi C (2018) Survey on human–robot collaboration in industrial settings: safety, intuitive interfaces and applications. Mechatronics (oxford) 55:248–266. https://doi.org/10.1016/j.mechatronics.2018.02.009

Weckenborg C, Kieckhäfer K, Müller C et al (2020) Balancing of assembly lines with collaborative robots. Bus Res 13:93–132. https://doi.org/10.1007/s40685-019-0101-y

Weckenborg C, Thies C, Spengler TS (2022) Harmonizing ergonomics and economics of assembly lines using collaborative robots and exoskeletons. J Manuf Syst 62:681–702. https://doi.org/10.1016/j.jmsy.2022.02.005

Wu E-F, Jin Y, Bao J-S, Hu X-F (2008) A branch-and-bound algorithm for two-sided assembly line balancing. Int J Adv Manuf Technol 39:1009–1015. https://doi.org/10.1007/s00170-007-1286-3

Yilmaz H, Yilmaz M (2020) A mathematical model and tabu search algorithm for multi-manned assembly line balancing problems with assignment restrictions. Eng Optim 52:856–874. https://doi.org/10.1080/0305215x.2019.1618288