Balancing a Robotic Spot Welding Manufacturing Line: an Industrial Case Study

Thiago Cantos Lopes[a], Celso Gustavo Stall Sikora[a], Rafael Gobbi Molina[b], Daniel Schibelbain[c], Luiz Carlos de Abreu Rodrigues[b], Leandro Magatão[a]*

a: Graduate Program in Electrical and Computer Engineering (CPGEI)

Federal University of Technology - Paraná (UTFPR), Curitiba, Brazil, 80230-901

b: Department of Mechanical Engineering - UTFPR

c: Renault do Brasil, São José dos Pinhais, Brazil, 83070-900

Abstract

Balancing robotic welding manufacturing lines is a challenging variation of assembly line problems, often performed manually and based exclusively on the company's experience. The combination of the problem's characteristics: movements, assignment restrictions, assignment-dependent parameters and interference constraints pose significant modeling and practical difficulties. The combination of this problem's features is rather difficult to properly convert to previously described models. In this paper, we describe the problem at hand, we present the model developed to solve it and describe the case studies in a real-world car factory on the outskirts of Curitiba in Brazil. The model was developed with Mixed Integer Linear Programming (MILP) techniques, validated with empirical data, and solved with a universal solver. The optimized balancing achieved a cycle time reduction of up to 6.6% compared to the as-is configurations. The total movement time was observed not to be necessarily minimized during the optimization process, implying that trade-offs exist between movement times and the number of welding points performed.

Keywords: Combinatorial Optimization, Assembly Line Balancing Problem, Robotic Welding Manufacturing Line, Mixed Integer Linear Programming

DOI: 10.1016/j.ejor.2017.06.001

1. Introduction

During the manufacturing of a vehicle's body, its different pieces are assembled and welded together. This process can be performed manually or with robots (or in a hybrid manner). In a sense, this can be seen as a classic balancing problem: The shop's cycle time is defined by the most loaded worker or robot; therefore, efficient balancing seeks to distribute the workload in an even manner. There are, however, a series of particular features in this problem, such as movements

*Corresponding author
Email address: magatao@utfpr.edu.br (Leandro Magatão[a])

and interferences. These features distance the problem from the classical assembly line balancing variants. In this paper we describe a new model and case study that seeks to optimize robotic manufacturing lines that perform the spot-welding points in vehicles.

Assembly Line Problems have a wide range of variations. Mostly, they derive from the simpler version, the Simple Assembly Line Balancing Problem (SALBP). These variations either modify SALBP's basic concepts or add new ones entirely. The reviews presented in Becker & Scholl (2006); Boysen et al. (2009); Battaïa & Dolgui (2013) provide broad classifications ranging from task and station attributes, line control and layout and model-dependency.

In a robotic welding context, one may try to adapt the problem at hand to fit models already described in the literature. For instance, some welding points can only be accessed by a subset of robots: these accessibility issues might be treated with assignment restrictions (Pastor & Corominas, 2000). Robotic Assembly Line Balancing Problems (Rubinovitz & Bukchin, 1991) usually attribute different processing times for tasks depending on which station the tasks are performed, or on which robot is assigned to each station. Sequence-dependence increments (Scholl et al., 2013) seem to be relevant to describe movements as the duration of one spot welding tends to be small in comparison to the cycle time (in the case studies, about 20 times shorter). Furthermore when there are many robots at the same workstation, one could attempt to treat interference constraints as incompatibility constraints between tasks. However, there are limitations to how closely one can adapt the many features of this problem to current models, in particular when the characteristics are considered in a combined fashion.

This paper presents an optimization problem, a novel MILP model and the case study developed in a large size spot welding robotic line. The paper is structured as follows. Section 2 presents the studied problem, describes its characteristics and explains the limits other models display when employed to describe and optimize it. Section 3 describes the developed model from its basic concepts to its sets, variables and constraints. Section 4 describes the case studies, the problem's size, achieved results and discusses some practical insights. Section 5 summarizes the main results and contributions of this paper.

2. Problem Statement

The studied manufacturing line is composed primarily of welding points. These can be seen as tasks whose required processing times are similar and short when compared to the cycle time. In principle, balancing might seem simple: simply divide the number of welding points roughly evenly between robots. There are, however, a few difficulties: robotic tools must be moved between welding points and the movement time is not usually known between each pair of points - and it is not necessarily proportional to euclidean distance between points, either. Such time cannot be easily determined as optimizing robotic routes is a hard problem on its own. Furthermore, stations often have more than one robot. Robots in the same station are subject to potential interference problems if they perform tasks (welding points) at the same regions or cross the robotic arms. In manual stations, this might not be such a serious concern as workers can communicate, coordinate and adapt in real time, but the studied robots blindly follow a predefined set of instructions. The specific tools and fixed positions of robots impose accessibility constraints, meaning that each robot will only be able to access and perform a subset of the welding points.

The studied robotic spot-welding balancing problem has, therefore, the following characteristics:

- C0 Occurrence Constraint - all welding points must be performed;
- C1 Assignment Restrictions - due to accessibility constraints associated with the robot's tools and positioning;
- C2 Robot-wise dependent parameters - Not all robots are equally fast due to their model and to the size of their spot welding tools
- C3 Movement Time Between Welding Points - Each robot's cycle time is not determined by the simple sum of time of each "task"; performed: movement times between points must be taken into account
- C4 Interference Constraints - robots must not collide with each other as they perform tasks;

While the characteristic C0 is simple and a basic part of nearly all assembly line models (Bowman, 1960; White, 1961; Thangavelu & Shetty, 1971; Patterson & Albracht, 1975; Scholl, 1999) it is made slightly more complicated by the characteristic C1, as explained hereafter. Figures 1 and 2 illustrate why the problem is assignment bounded, as each robot can assess only a subset of welding points, due to geometrical aspects. These assignment constraints can be seen as station-type restrictions in the classification schemes proposed by Boysen et al. (2007, 2008); Battaïa & Dolgui (2013). In Figure 1, the robot can reach the welding point without colliding with the vehicle. In Figure 2, the robot will collide with the vehicle, if it attempts to reach the welding point.

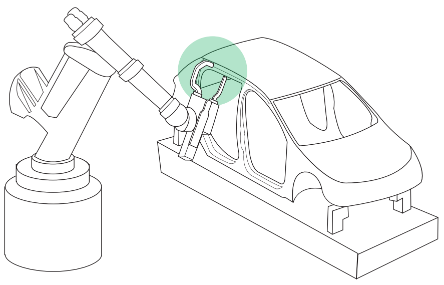
Figure 1: Accessible welding point.

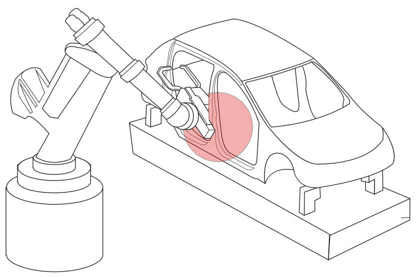
Figure 2: Inaccessible welding point.

There are models (Scholl et al., 2010) and algorithms that deal with various assignment restrictions (Dar-El & Rubinovitch, 1979; Pastor & Corominas, 2000; Lapierre et al., 2004; Bautista & Pereira, 2007) and, thus, could be used to deal with C1. C2 is a pretty common part of robotic balancing problems (first defined by Rubinovitz & Bukchin (1991)). The problem would probably be treatable by assignment bounded or adapted RALBP algorithms (Kim & Park, 1995; Levetin et al., 2006; Daoud et al., 2014), if C1 and C2 were the problem's most challenging characteristics. But they are not. C3 implies on constraints that make this problem harder to translate into a SALBP-based model, as it violates a core hypothesis of such models: a station's (robot's) time is not equal to the sum of the task's time performed in it. This fact happens because the welding time is comparable to the movement's time between points. Figures 3 and 4 show how relatively close welding points might require absolutely different configurations for the robot: this fact implies that movements are not only a matter of Euclidean distance.

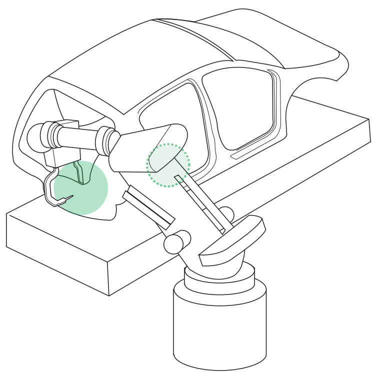
Figure 3: Robot performing a welding point in the back of the vehicle, before moving to a nearby welding point.

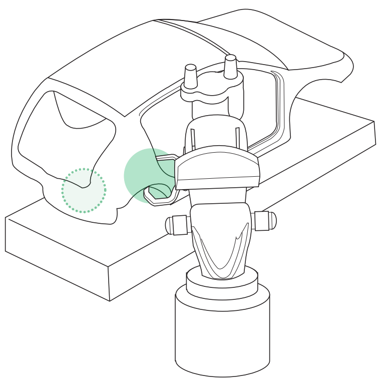
Figure 4: Robot performing a welding point close to the first one (Figure 3). Notice how the robot's position has changed significantly.

There are set-up models (described for instance by Scholl et al. (2013)) and algorithms (Andrés et al., 2008; Scholl et al., 2008; Martino & Pastor, 2010) that could attempt to describe C3 by comparing movement times to setup times. But these models and algorithms require a matrix of known set-up times between tasks. Such data is not available as the time to move between each pair of welding spots is rather difficult to determine, especially given the large number of welding points involved. We could attempt to use an approximation for such movement times, but such times would have to be robot-dependent. The line has robots of different models with different speeds and tools of different sizes, aka C2. This might not be a promising direction though.

Attempting to adapt the problem at hand to the model described by Scholl et al. (2013), the number of variables and constraints would tend to be too large: In the studied scenarios, there were 42 robots. Assuming each of the roughly 700 welding points can be accessed by 10% of the robots (rounding down to 4) and that the points can be followed or preceded by about a third of the other points (rounding down to 200), the sequence variables (y_{ij} and w_{ij}, see Scholl et al. (2013) for more details) would amount to 280.000 binaries and would require nearly half a million constraints to be properly controlled. Such a MILP problem is almost certain to be computationally un-treatable. And that is setting aside the fact that the set-up times are robot-dependent. Successfully operating some simplifications, there are heuristics and algorithms that could, maybe, be able to provide good answers, were it not for the other characteristics of this problem.

The feature C3 is made even harder by the fact that some welding points can be performed in different manners, by different accesses for the robotic arms. The manner a welding point is accessed affects the movement time the robot will take to reach other points. Two points may seem really close and there might be a direct route between them but only if they are performed with the robotic arm positioned in a way that such route is feasible. These characteristics are illustrated by Figures 5 and 6, where the robot is accessing the same welding point from two different directions: suppose the robot is to move to a welding point between the doors. Such point is reachable for the robot in Figure 6 with a simple spin of its tool. The robot in Figure 5, in the other hand, must first leave the windshield region, then spin while moving inside the front door (without colliding with the vehicle), and only then it reaches the welding point. A symmetrical situation occurs if the robot were to move to a welding point in the central lower part of the windshield region: this time would be easier for the robot in Figure 5 move between the points. This means that a set-up time approximation to movements between welding points would not only be *robot-dependent* but also *access-dependent*, making the use of a set-up based models and algorithms even more challenging. Parallels with assembly lines with processing alternatives models (Pinto et al., 1983) can be drawn, but the nature of the alternatives is, in this case, rather different.

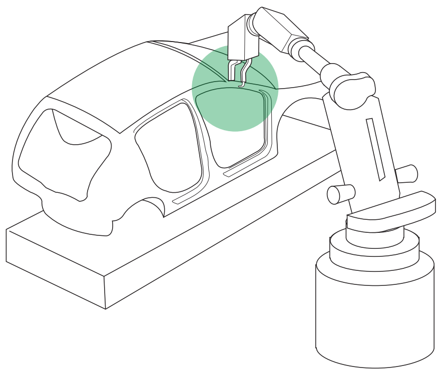
Figure 5: Welding point accessed through the windshield.

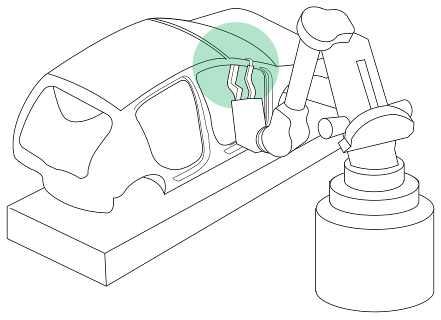
Figure 6: Welding Point accessed through the front door.

But even if we could overcome this process alternative aspect and treat C0, C1, C2 and C3 using set-up based assignment bounded robotic balancing model, there is C4: Robots must not collide with one another while performing the welding points. The interference constraints are not translatable into the constraint types described by Boysen et al. (2007, 2008); Battaïa & Dolgui (2013). They are not merely a function of the number of welding points in a station. For instance if one robot performs part of the welding points in the door while the other performs the rest of the points in the same door in the same station, as in Figure 8: Under certain conditions it might be possible to sequence the robots and avoid interference, but not always. Although the time robots spend at each access (say the front door) can be seen as a scarce resource, this resource has not got a fixed amount. When more than one robot have to use the same access at the same station, the time required for switching robots decreases the amount of "resource" available. This is very different, however, from other formulations usually classified under cumulative restrictions of task-station assignments (Boysen et al., 2007), as two or more stations (robots) share a non-fixed amount of a resource. For the sake of comparison, Bautista & Pereira (2007) employ space constraints that are defined station-wise. It is clear that multiple robots per station do not offer the same possibilities as multiple stations with single robots. Contrary to most works with station parallelism (first studied by Buxey (1974)), these robots often do not offer space synergy, but might interfere with each other.

Another interference (C4) example lies in robots crossing arms (Figure 7). It is possible for *either* the front robot perform welding points in the back of the car, *or* the back robot perform welding points in the front of the car. However, if they do so simultaneously the chances of collision of the robotic arms are great, as depicted by the Figure 7, where both robots perform accessible welds, but the station's configuration is undesirable.

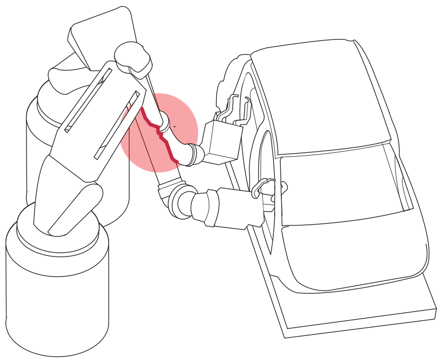
Figure 7: Arm Crossing Interference.

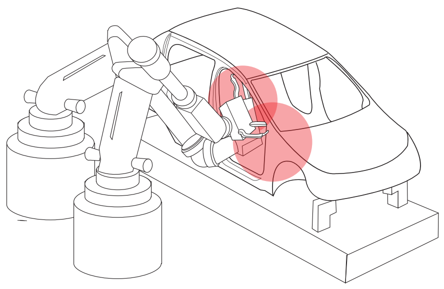
Figure 8: Space Dispute Interference.

These interference characteristics are very difficult to translate into adapted features of the classification schemes of Becker & Scholl (2006), Boysen et al. (2007) and Battaïa & Dolgui (2013). Consequently, the combination of all the problem's features give rise to a very complex ALB problem that is even more difficult to adapt into previously described models. Prior to the work described in this paper, the studied manufacturing line's balancing was performed manually, based almost exclusively on the engineer's and operator's experiences. To the best of our knowledge, there is no literature model that can describe all these characteristics in a single model. This means that balancing such manufacturing line requires a new model.

3. Model

3.1. General Concepts

The employed model operates some simplifications in order to approximate the real-world problem. The main simplification is that we do not establish a sequence for points performed by each robot, but rather count the number of points and the number of movements (classified as: very small, small, medium and large) required to perform said points. This simplification is based on two observed facts: Firstly, most of the welding points were distributed in logical paths such that the robot's time spent on that path was proportional to the number of points performed. Secondly, when the robot moved from one path to another, the time spent on such movement could usually be classified into one of four classes, which are further discussed at the end of this section.

Robots are presumed to be fixed to a station, with fixed given properties and parameters (such as speed, tools, position and capabilities). Two parameters of interest were the relative velocities of each robot and the fixed time required per welding point. This robot-wise fixed time is associated with opening and closing the welding claws. In general, newer robots tend to allow higher speeds due to better technological features and robots with bigger tools (often required to access some welding points) tend to have lower speeds and longer fixed times due to slower operation. All stations have a fixed constant time associated with the entrance and exit of each vehicle of said station, time during which it is assumed that no welding point can be performed.

Welding points are grouped in regions, as depicted by the Figure 9 in which welding points (red dots) are boxed in accordance with physical characteristics and the accessibility constraints. Regions have the following characteristics:

- All welding points in a region take the same amount of time to be performed. This time is added to fixed time per point of the robot that performs it. The movement time between points of the same region can be regarded as part of the time required to perform a welding point;
- The time spent by a robot in a region is proportional to the number of points performed by said robot at said region;
- If a robot can access the region, it can access all points in it;
- Each region r has a fixed and given number of points N_r;
- In a first approximation, the movement time is considered constant between regions. That is: if a robot works at three regions, it will move twice between regions and have a movement time twice as great as a robot that only works at two regions (and thus only moves once between regions);

The region hypothesis was based on a problem's characteristic: welding points were mostly lined up in logical geometrical paths and locations, and these logical paths could be divided in accordance to how the robots could access them.

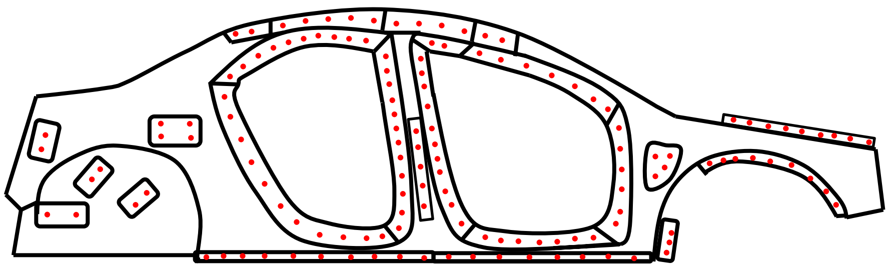

Figure 9: Example on how welding points were grouped in regions, adapted from the case study.

Welding points can be performed from different accesses: front door, back door, luggage compartment, windshield. Some regions can be reached by multiple accesses (Figures 5 and 6). In order to better describe movements, accesses must be taken in account. If a robot is accessing regions within the same access, then the movement time tend to be smaller than when the robot accesses regions from different accesses. The concept of accesses adds a different kind of movement to the problem. Movements between accesses are analogous to movements between regions and can be analogously counted. Figure 10 illustrates the different regions reachable by accesses: some points can only be performed with the robotic arm positioned in a certain access, for instance green indicate regions accessible through the back door, while blue indicates regions accessible through the front door.

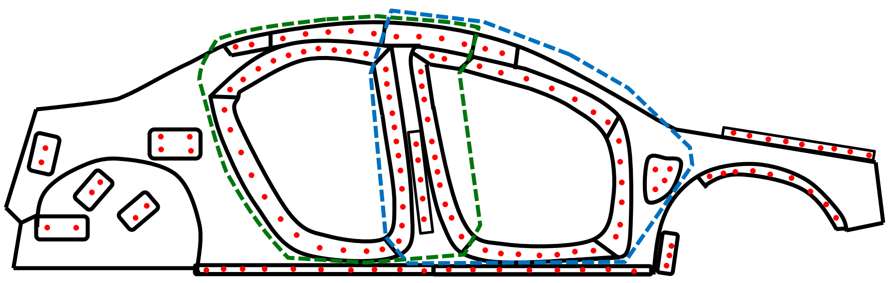

Figure 10: Accesses for the regions and adjacencies

Further movement time refinements are provided by the adjacency concept: Two regions that are next to one another (one is connected to the other) are deemed adjacent, meaning that in some cases the movement between them can be ignored or reduced. For instance, the regions that surround the front door can be seen as adjacent (as shown in Figure 10). The same applies to accesses. The movement time between the front and back doors is naturally smaller than the movement time between the front door and the luggage compartment. In many cases a large physical region must be divided in several adjacent regions because of accessibility constraints: if a robot can access the region, it can access all the welding points in it. Therefore, regions were modeled according to both accessibility and movements. Adjacencies were used to better describe movement times. For instance, in Figure 9, the front door's cycle of regions is composed of adjacent regions, but they must be divided in order to account for the accessibility capacities of different robots.

Some constraints required grouping regions in sets that share some particular characteristics. There are two types of groups of regions that must be taken in consideration: Macro-regions (used to deal with interference constraints) and cycles (used to prevent incorrect counting of the number of adjacencies a robot has benefited from). The particular way in which these groups of regions behave is described in the Section 3.3. The Figure 11 illustrates a pair of possible crossing constraint macro-regions: if, in the same station, the front robot accesses the back macro-region and the back robot accesses the front region they are likely to collide their robotic arms, as illustrated by the Figure 7.

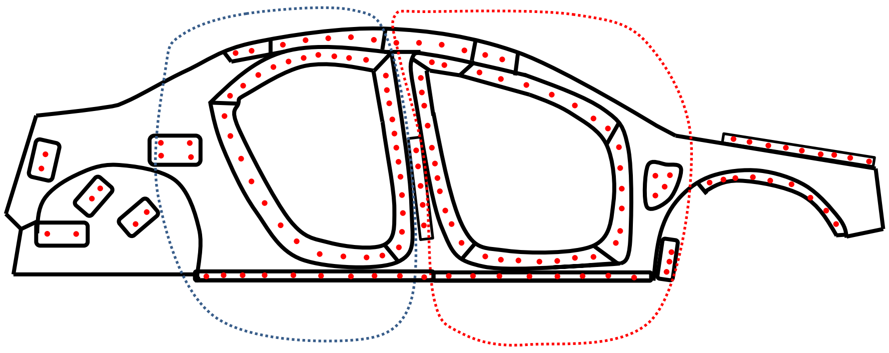
Figure 11: Macro-regions defined for interference constraints

As previously mentioned, the number of movements is counted. Movements between regions within the same access are deemed "small" (or "very small" if the regions are adjacent). Movements between accesses are deemed "large", unless there is an adjacency between the accesses (in which case they are "medium" movements). Here are examples of each movement type: Small - Moving from the left to the right side of the front door. Medium - Moving from the front door to the back door. Large - moving from the front door to the luggage compartment.

Movements are counted by adding the number of regions and accesses a robot performs welding points, and subtracting the time-wise difference linked to the number of allowable adjacencies. The restrictions that control these operations will be described in Section 3.3.

3.2. Sets and Variables

The problem is modeled around its decision variables, that are all defined based on parameter sets. In order to differentiate sets of parameters and sets of variables, the later will start with a lower-case letter (v for integer and real variables and b for binary variables). The tuple sets that are employed to define the variables are a discrete collection of tuples, whose interpretations are summarized at the Table 1. The problem's variables, their types, tuple sets and interpretation are listed and summarized at the Table 2. Lastly, some constraints require some additional parameters that are either vectors or constants. These parameters are listed by the Table 3.

Each variable set is based on a domain (tuple sets) and has one variable for each element of said set. For example, the variable set bWA has one binary variable for each (w, a) tuple in the tuple set WA, namely $bWA_{(w,a)}$. The tuple and variable sets mostly follow a concept-based letter orientation. For example, R stands for regions, W for robots, A for accesses, S for stations. This pattern seeks to simplify interpretation: for instance, the binary variable $bWA_{(w,a)}$ is set to true when the robot w performs welding points by the access a.

All variables in the proposed model are strictly non-negative. In our case the main decision variable is an integer number of welding points, performed by the robot w at the region r via the access a. The variable set $vWRA$ contains one integer variable for each element of the tuple set

Table 1: Parameter Tuple sets: They are employed throughout the model from the variable definitions to the constraint formulations

Sets	**Tuple**	**Description of each range and meaning of the tuples on each set**
S	s	Range for all stations
R	r	Range for all regions
W	w	Range for all robots
WS	(w, s)	Indicates that the robot w is allocated at the station s
WRA	(w, r, a)	Indicates that the robot w can reach the region r via the access a
WA	(w, a)	Indicates that the robot w can use the access a
RR_{adj}	(r_1, r_2)	Indicates that the regions r_1 and r_2 are adjacent
WRR_{adj}	(w, r_1, r_2, a)	Indicates that the robot w can benefit from the adjacency between r_1 and r_2 via the access a
WAA_{adj}	(w, a_1, a_2)	Indicates that the robot w can benefit from the adjacency between a_1 and a_2
CR	(c, r)	Indicates that the region r belongs to the region-cycle c
CA	(c, a)	Indicates that the access a belongs to the access-cycle c
WCR	(w, cr)	Indicates that the robot w can perform the region-cycle cr
WCA	(w, ca)	Indicates that the robot w can perform the access-cycle ca
WEM	(w, em)	Indicates that the robot w can access the exclusion macro-region em
WCM	(w, cm)	Indicates that the robot w can access the crossing macro-region cm
EMR	(em, r)	Indicates that the region r belongs to the exclusion macro-region em
CMR	(cm, r)	Indicates that the region r belongs to the crossing macro-region cm
$WMWM$	(w_1, m_1, w_2, m_2)	Indicates that the robot w_1 cannot access the crossing macro-region m_1 at the same time that robot w_2 accesses the crossing macro-region m_2
EA	ea	Indicates that the access a defines a exclusion interference constraint
EM	em	List of all the macro regions em that define exclusion constraints

Table 2: Variables sets, their types, the sets and tuples they are defined by and their meaning

Variable Set	**Tuple Set**	**Tuple**	**Description and meaning of the variable from the set**
vCT	-	-	Line's cycle time (Time Units)
$vCTW$	W	w	Robot w's cycle time (Time Units)
$vWRA$	WRA	(w, r, a)	Number of welding points performed by robot w on region r via the access a
$bWRA$	WRA	(w, r, a)	Whether or not the robot w performs points on region r via the access a
$vNMR$	W	w	Number of movements between regions performed by the robot w
$vNMA$	W	w	Number of movements between accesses performed by the robot w
$bWRR$	WRR_{adj}	(w, r_1, r_2, a)	Whether or not the robot w benefits from the adjacency between the regions r_1 and r_2 via the access a
$bWAA$	WAA_{adj}	(w, a_1, a_2)	Whether or not the robot w benefits from the adjacency between the accesses a_1 and a_2
bWA	WA	(w, a)	Whether or not the robot w uses the access a
$bWEM$	WEM	(w, m)	Whether or not the robot w performs points on the exclusion macro-region m
$bWCM$	WCM	(w, m)	Whether or not the robot w performs points on the crossing macro-region m

Table 3: Parameters employed for occurrence, time controlling and interference constraints

Parameters	**Set**	**Tuple**	**Meaning**
T_s	-	-	Fixed time associated to preparation at stations (Time Units)
N_r	R	r	Number of points in the region r
T_r	R	r	Duration per point in the region r (Time Units)
T_w	W	w	Duration per point for the robot w (Time Units)
V_w	W	w	Relative Speed of the robot w
T_{movR}	-	-	Movement time between regions (Time Units)
T_{movA}	-	-	Movement time between accesses (Time Units)
T_{adjR}	-	-	Time gained per adjacency between regions (Time Units)
T_{adjA}	-	-	Time gained per adjacency between accesses (Time Units)
U_p	-	-	Upper bound parameter for exclusion constraints
D_p	-	-	Decrease per robot parameter for exclusion constraints

WRA of accessibility (that contains the tuples $(w,\ r,\ a)$ of regions r accessible by the robot w via the access a). This is a heavily assignment bounded problem: in the studied cases only about 10% of the possible $(w,\ r,\ a)$ combinations were feasible.

There are many decision variables that follow $vWRA$, the first one is the binary $bWRA$. Each $bWRA_{(w,r,a)}$ that is true when its corresponding $vWRA_{(w,r,a)} \geq 1$, and false otherwise. The variables in the set bWA are true if the worker w uses the access a (WA is another accessibility set).

Movement variables are defined for each w in W: $vNMR$ counts the number of movements between regions and $vNMA$ counts the number of movements between accesses. The adjacency variables $bWRR$ ($bWAA$) are true when the robot w benefits from adjacencies between the regions r_1 and r_2 (accesses a_1 and a_2) and are defined for each tuple $(w,\ r_1,\ r_2)$ in WRR_{adj} ($(w,\ a_1,\ a_2)$ in WAA_{adj}) of feasible adjacency.

The last parameter sets for movements are CR and CA, whose tuples $(c,\ r)$ (and $(c,\ a)$) indicate that the region r (the access a) is part of the region cycle c (the access cycle c). Their main purpose is to limit the number of adjacencies a robot can benefit from. They are defined when there are adjacencies that form a cycle, in which case it is possible to have more adjacencies than movements. As shown by Figure 12, a cycle with 4 adjacent regions has 4 adjacencies and 3 movements. This is why cycles are part of the logical constraint of adjacencies between regions and accesses. The set WCR (WCA) indicates that the robot w can perform welding points in all regions (accesses) of the region (access) cycle cr (ca), and is a set that can be determined by the accessibility sets WRA and WA.

Finally, the interference constraints require two sets of macro-regions: exclusion and crossing macro-regions. Their parameters are sets EMR and CMR, with tuples $(m,\ r)$ that list regions r that belong to each macro-region m. The variable set $bWEM$ ($bWCM$) contains binary variables for each tuple $(w,\ m)$ in the set WEM (WCM) with macro-regions m where the robot w can perform welding points. These variables indicate whether or not the robot performs welding points in the macro-region. The crossing constraints also require crossing parameter tuples $(w_1,\ m_1,\ w_2,\ m_2)$ (set $WMWM$) that inform that if the robot w_1 accesses the macro-region m_1

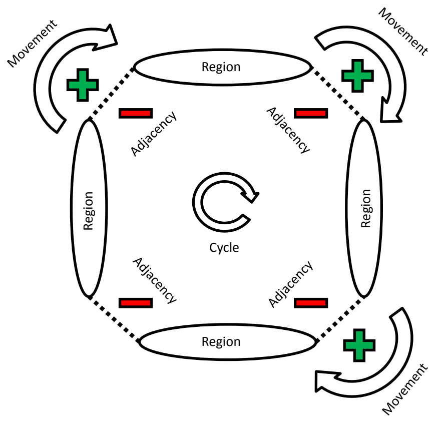

Figure 12: Illustration of the reason cycles are employed to correctly count the number of movements and adjacencies: If a robot displaces through regions that form an adjacency cycle, there will be one more adjacency than the number of movements, this implies that the adjacency time benefits could be higher than the time spent on the required movements. Cycle constraints (Inequalities 12 and 13) prevent such wrong consideration.

and the robot w_2 accesses the macro-region m_2 they may collide, therefore, leading to an unfeasible solution. Another variant of the exclusion constraint formulation is the access-wise interference constraint, defined based on the number of robots that employ the same access a in the exclusion access set EA.

The last variables within the proposed model are the cycle time of the manufacturing line (vCT) and the cycle time of each individual robot w ($vCTW_w$).

3.3. Constraints

There are three kinds of constraints in this model. Firs, the occurrence constraint requires that all welding points in all regions be performed. Second, there are the time measuring constraints that seek to determine the cycle time of each robot and of the line as a whole. Lastly, the interference constraints seek to ensure that robots do not collide during the performance of their tasks.

3.3.1. Occurrence Constraint

The occurrence constraint states that the sum of welding points performed by all robots by all accesses in each region equals the number of welding points in the region, as stated by the Equation 1. This constraint is defined based on the assignment binding WRA set that allows only robots that can access a region to perform points in it.

$$\sum_{(w,\ r,\ a) \in WRA} vWRA_{(w,\ r,\ a)} = N_r \qquad \forall\, r \in R \tag{1}$$

3.3.2. Time Controlling Constraints

The cycle time (vCT) is determined by the line's bottleneck station, worker or, in this case, robot ($vCTW_w$). This is stated in the Inequality 2.

$$vCT \ \geq \ vCTW_w \qquad \forall \, w \in W \tag{2}$$

The cycle time of each robot w ($vCTW_w$) is determined by the Equation 3. The total time of a robot is given by the time required to perform the welding points assigned to it added to the movement time and to the preparation time T_s of its station. The movement time is estimated as the sum of movements between regions (multiplied by the time between regions T_{movR}), added to the sum of movements between accesses (idem with T_{movA}) and subtracted by the sum of adjacency time gains between regions and between accesses (with times T_{adjR} and T_{adjA}).

$$\begin{aligned} vCTW_w \ = \ & \sum_{(w,\ r,\ a) \in WRA} vWRA_{(w,\ r,\ a)} \cdot (T_w + T_r) && + \\ & + \ T_{movR}/V_w \ \cdot \ vNMR_w && + \\ & + \ T_{movA}/V_w \ \cdot \ vNMA_w && - \\ & - \ T_{adjR}/V_w \ \cdot \sum_{(w,\ r_1,\ r_2,\ a) \in WRR_{adj}} bWRR_{(w,\ r_1,\ r_2,\ a)} && - \\ & - \ T_{adjA}/V_w \ \cdot \sum_{(w,\ a_1,\ a_2) \in WAA_{adj}} bWAA_{(w,\ a_1,\ a_2)} \ + T_s && \forall \, w \in W \end{aligned} \tag{3}$$

The model's variables that control each part of the robot's movement approximation are described in the following equations. Firstly, the variables in the sets $bWRA$ and bWA are controlled by the Inequalities 4 and 5. These variables determine in which regions and accesses each robot operates. These constraints establish a fixed charge aspect to the problem: the movement costs of accessing regions. This occurs both in regard to the number of points in a region and in regard to the number of regions in an access (Hillier & Lieberman, 2015). As such, it is expected that good solutions tend to have fewer movements, concentrating each robot's work in areas closer to one another.

$$bWRA_{(w,\ r,\ a)} \ \geq \ vWRA_{(w,\ r,\ a)}/N_r \qquad \forall \, (w,\ r,\ a) \ \in \ WRA \tag{4}$$

$$bWA_{(w,\ a)} \ \geq \ bWRA_{(w,\ r,\ a)} \qquad \forall \, (w,\ r,\ a) \ \in \ WRA \tag{5}$$

The first movement variables defined are the movement counters: the number of movements between regions ($vNMR_w$) or accesses ($vNMA_w$) is the number of regions or accesses minus 1 (unless the robot works in no region (or access) in which case the number of movements is zero). This information is codified in the Inequalities 6 and 7 (as stated in Section 3.2, all variables are strictly non-negative).

$$vNMR_w \ \geq \ -1 + \sum_{(w,\ r,\ a) \in WRA} bWRA_{(w,\ r,\ a)} \qquad \forall \, w \ \in \ W \tag{6}$$

$$vNMA_w \ \geq \ -1 + \sum_{(w,\ a) \in WA} bWA_{(w,\ a)} \qquad \forall \, w \ \in \ W \tag{7}$$

The adjacency variables can only be set as true if the robot w works at both regions $r_1,\ r_2$ (accesses $a_1,\ a_2$) on the set WRR_{adj} (WAA_{adj}). This restriction is stated by the Inequalities 8a and 8b (Inequalities 9a and 9b). The adjacency gain between regions can only happen if both regions are accessed by the same access a..

$$bWRR_{(w,\ r_1,\ r_2,\ a)} \leq bWRA_{(w,\ r_1,\ a)} \qquad \forall\ (w,\ r_1,\ r_2,\ a)\ \in WRR_{adj} \tag{8a}$$

$$bWRR_{(w,\ r_1,\ r_2,\ a)} \leq bWRA_{(w,\ r_2,\ a)} \qquad \forall\ (w,\ r_1,\ r_2,\ a)\ \in WRR_{adj} \tag{8b}$$

$$bWAA_{(w,\ a_1,\ a_2)} \leq bWA_{(w,\ a_1)} \qquad \forall\ (w,\ a_1,\ a_2)\ \in WAA_{adj} \tag{9a}$$

$$bWAA_{(w,\ a_1,\ a_2)} \leq bWA_{(w,\ a_2)} \qquad \forall\ (w,\ a_1,\ a_2)\ \in WAA_{adj} \tag{9b}$$

Region-to-region adjacencies require some limitations: Each adjacency between two regions r_1 and r_2 (listed by $(r_1,\ r_2)$ tuples in RR_{adj}) can only be used by one robot (Inequality 10). Furthermore, each robot w can only benefit from two adjacencies tied to a given region r (Inequality 11). These constraints are required to prevent mathematical gains with no real-world counterpart.

$$\sum_{(w,\ r_1,\ r_2)\in WRR_{adj}} bWRR_{(w,\ r_1,\ r_2)} \leq 1 \qquad \forall\ (r_1,\ r_2) \in RR_{adj} \tag{10}$$

$$\begin{aligned} &\sum_{(w,\ r,\ r_1)\in WRR_{adj}} bWRR_{(w,\ r,\ r_1)} + \\ &+ \sum_{(w,\ r_2,\ r)\in WRR_{adj}} bWRR_{(w,\ r_2,\ r)} \leq 2 \qquad \forall (w,\ r,\ a) \in WRA \end{aligned} \tag{11}$$

Lastly, the cycle constraints limit adjacencies region-wise (CR contains tuples $(c,\ r)$ of adjacent regions r that form a cycle c) and access-wise (CA, idem as regions with $(c,\ a)$ for accesses that form a cycle), as stated by the Inequality 12 (Inequality 13). These constraints are relevant for robots w that can access all regions r in the cycle c: even if it performs welding points in all regions of said cycle, it cannot benefit from all adjacencies.

$$\sum_{\substack{(c,\ r_1)\ \in\ CR \\ (c,\ r_2)\ \in\ CR \\ (w,\ r_1,\ r_2,\ a)\ \in\ WRR_{adj}}} bWRR_{(w,\ r_1,\ r_2,\ a)} \leq -1 + \sum_{(c,\ r)\ \in\ CR} 1 \qquad \forall\ (w,\ c)\ \in WCR \tag{12}$$

$$\sum_{\substack{(c,\ a_1)\ \in\ CA \\ (c,\ a_2)\ \in\ CA \\ (w,\ a_1,\ a_2)\ \in\ WAA_{adj}}} bWAA_{(w,\ a_1,\ a_2)} \leq -1 + \sum_{(c,\ a)\ \in\ CA} 1 \qquad \forall\ (w,\ c)\ \in WCA \tag{13}$$

3.3.3. Interference Constraints

Four kinds of interference constraints were considered: region station-wise exclusivity, crossing-macro regions, exclusion macro-regions, and access-wise exclusion. They seek to describe problems that may happen as robots "compete" for the same physical space.

The region station-wise exclusivity states that at each station s, at most one robot w can access each region r. This constraint is stated by the Inequality 14. Note that this constraint prevents a robot from accessing the same region from two different accesses. From a practical standpoint, this is a desirable feature as that solution would be dominated by the one in which the robot performs the same number of points via only one access.

$$\sum_{\substack{(w,\ s) \in WS \\ (w,\ r,\ a) \in WRA}} bWRA_{(w,\ r,\ a)} \ \leq\ 1 \qquad \forall\ s\ \in\ S,\ r\ \in\ R \tag{14}$$

The macro-region allocation variables control the two types of macro-region interference constraints. A robot is allocated to a macro-region if it performs welding points in one of the regions of said macro-region. This is stated by the Inequalities 15a and 15b.

$$bWEM_{(w,m)} \geq bWRA_{(w,\ r,\ a)} \quad \forall\ (w,\ r,\ a) \in WRA, (m,\ r) \in EMR \tag{15a}$$

$$bWCM_{(w,m)} \geq bWRA_{(w,\ r,\ a)} \quad \forall\ (w,\ r,\ a) \in WRA, (m,\ r) \in CMR \tag{15b}$$

The crossing macro-region constraints seek to forcefully ban two different robots r_1 and r_2 from occupying conflicting physical space defined by the macro-regions m_1 and m_2 (notice that m_1 and m_2 might be the same, but the robots must be different). In essence at most either r_1 occupies m_1 or r_2 occupies m_2. This constraint is codified by the Inequality 16. For instance, suppose that w_f is the front robot and w_b is the back robot, while m_f and m_b are the front and back door macro-regions. In this case $(w_f,\ m_b,\ w_b,\ m_f)$ impedes the crossing of the robot arm illustrated by the Figure 7: It allows either the front robot to work in the back of the car or (exclusive or) the back robot to work at the front of the car.

$$bWCM_{(w_1,\ m_1)}\ +\ bWCM_{(w_2,\ m_2)}\ \leq 1 \qquad \forall\ (w_1,\ m_1,\ w_2,\ m_2)\ \in WMWM \tag{16}$$

Ideally, one might want each overall region of the product to be occupied by at most one robot per station (and, thus, employ crossing macro-regions for all possible interference zones). If that is the case one could use crossing macro-region constraint with the same macro-region m for two or more robots. In this case $(w_f,\ m_f,\ w_b,\ m_f)$ would prevent two robots from simultaneously accessing the front door, as in Figure 8. But in some cases, due to accessibility constraints, this might simply be impossible or lead to a poor solution. Exclusion macro-regions are less restrictive interference constraints that seek to allow two robots to be assigned to the same zone in a way that they will together spend less than the cycle time in said zone. In this case, because the cycle time correlates very well with the number of welding points performed, it is stated that the sum of points performed will be limited by an upper-bound value (parameter U_p) that decreases (by the parameter D_p) for each robot that works at the same exclusion macro-region. This decrease happens because robots take time to enter and leave the access. This is stated by the Inequality 17.

This constraint that can be seen as an adaptation of resource constraints, as described on Section 1.

$$\sum_{\substack{(w,\ r,\ a)\in WRA \\ (w,\ s)\in WS \\ (m,\ r)\in EMR}} vWRA_{(w,\ r,\ a)} \ \leq\ U_p - D_p \cdot \left(-1 + \sum_{\substack{(w,\ s)\ \in\ WS \\ (w,\ m)\in WEM}} bWEM_{(w,\ m)} \right) \quad \forall\, s \in S,\ m\ \in\ EM \tag{17}$$

The goal of Inequality 17 is to limit the number of points performed in a macro region by a decreasing limit: the higher the number of robots in the same station performing welding points in the same macro-region, the smaller the total number of points they can perform. For instance, in the case studies of section 4, the upper bound value of number of the points (U_p) was 24. If two robots work at the same station in the same macro-region this limit decreases to 20 (therefore, D_p equals 4), as time will be required for one robot to leave the macro-region and for the other robot to enter it. These parameters were defined based on observation.

The access exclusion constraint is defined by the Inequality 18. This constraint is very similar to the macro-region exclusion constraint defined by Inequality 17. The main change is that the number of points performed on a same station by different robots via the same access (rather than macro-region) decreases the more robots simultaneously access said access-region.

$$\sum_{\substack{(w,\ r,\ a)\in WRA \\ (w,\ s)\in WS}} vWRA_{(w,\ r,\ a)} \ \leq\ U_p - D_p \cdot \left(-1 + \sum_{(w,\ a)\in WA} bWA_{(w,\ a)} \right) \quad \forall\, a \in\ EA, s\ \in\ S \tag{18}$$

4. Case Studies

The optimization models were developed based on one of the robotic spot welding lines of Renault's facility in the outskirts of Curitiba in Brazil. The line is depicted in a simplified manner by the Figure 13. It is composed of 42 robots distributed symmetrically (left and right sides of the vehicle) amongst 13 stations. Each stations had either two, four or six robots. When this model was developed, there were four car models (vehicle types) that passed through this manufacturing line, each of them with over 700 welding points. This was the last part of the car body manufacturing line, the body-in-white stage: the pieces were all assembled and, therefore, there were no precedence relations to deal with.

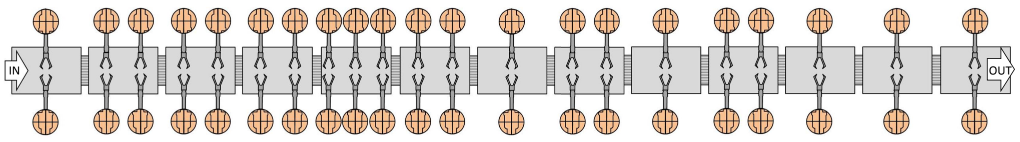

Figure 13: Studied line's simplified layout, with 42 robots allocated to 13 stations

The line was a mixed-model one with asynchronous pace, however, the total workload of each vehicle was very similar and their individual cycle times were not significantly different. This was partly due to the company's previous divisions of welding points, which were performed in such a way that left this line with a rather similar workload for all car models. Although in other parts of the manufacturing line sequencing could be a significant problem (see Boysen et al. (2009) for a mixed-model sequencing survey), in this part each vehicle could be balanced individually, allowing four (simpler) single model optimizations to replace a much more complex mixed-model one. This reduction of a Mixed-Model ALBP into P-Single-Model ALBPs could have downsides in manual lines due to the lack of specialization effects and set-up times (Becker & Scholl, 2006). The robots do not display learning curves and can easily alternate between products, allowing us to set these considerations aside. The main mixed-model aspect remaining was the accessibility limitations, defined based on the geometrical features of both: the robot and the vehicle models. However, this was considered an input rather than a variable. The goal was set to the minimization of cycle time of each product model. Therefore, one main instance was created for each vehicle.

4.1. Instances Definition and Data Calibration

The process of defining instances was not trivial: Firstly, data was collected as for where were the welding points. Secondly, the model's parameters (such as movement times, fixed times per robots) were determined by filming the robot's activities and carefully observing the videos. The model was designed to be slightly conservative, and, therefore, when a parameter (say movement time between non-adjacent regions of the same access) was observed with different values in different cases, the worst case was picked.

In order to assure that estimated task times are close to real times, a calibration must be performed. Figure 14 shows a comparison between the model's predictions and the robot's real times after the due data calibration for one instance. The model's approximation on the station workload (measured in time units) is higher than the real workload, indicating that any solution obtained by the proposed model will be an upper bound to the real solution.

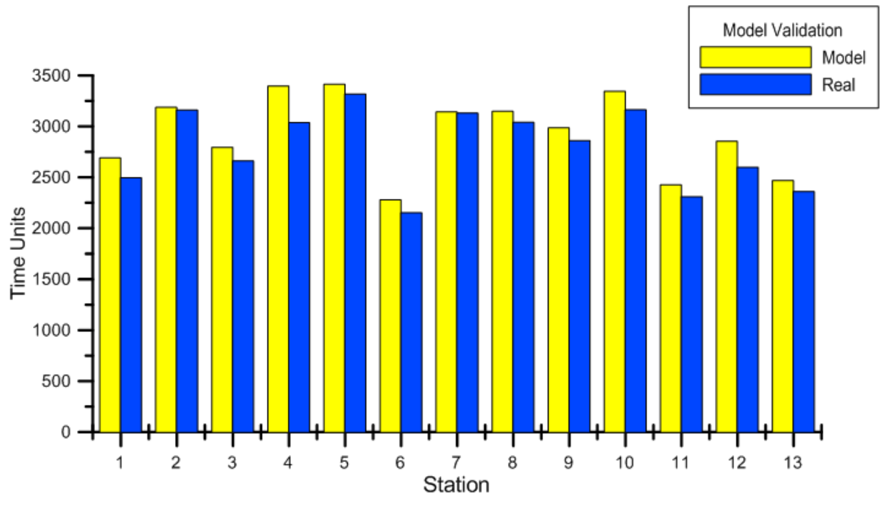

Figure 14: Comparison between the model's approximation to the empirically measured time: Notice that the approximations are slightly conservative

Table 4: Number of welding points (N), upper bounds (UB) and lower bounds (LB) on cycle time, and computation times for each instance. Each variant represent a product type. The 100% instances represent the practical case for each product.

	Variant 1				Variant 2			
% Points	N	UB	LB	time	N	UB	LB	time
10%	73	1,026.0	1,026.0	0s	68	972.0	972.0	74s
20%	146	1,439.5	1,439.5	2s	146	1,452.0	1,452.0	2s
30%	219	1,578.0	1,578.0	3s	221	1,564.4	1,564.4	4s
40%	287	1,745.5	1,745.5	3s	295	1,782.0	1,782.0	4s
50%	366	1,956.0	1,956.0	7s	372	1,980.0	1,980.0	4s
60%	459	2,208.0	2,208.0	34s	426	2,074.4	2,074.4	25s
70%	516	2,357.5	2,357.5	4s	518	2,334.0	2,334.0	19s
80%	591	2,561.5	2,561.5	13s	591	2,515.6	2,515.6	53s
90%	642	2,712.0	2,712.0	9s	640	2,664.0	2,664.0	5s
100%	733	2,867.5	2,867.5	45s	740	2,868.0	2,838.0	3600s
	Variant 3				Variant 4			
% Points	N	UB	LB	time	N	UB	LB	time
10%	71	1,075.6	1,075.6	2s	70	1,075.6	1,075.6	2s
20%	142	1,404.0	1,404.0	4s	142	1,422.0	1,422.0	6s
30%	217	1,620.0	1,620.0	9s	216	1,620.0	1,620.0	10s
40%	276	1,680.0	1,680.0	456s	273	1,713.3	1,713.3	247s
50%	348	1,986.0	1,986.0	48s	347	1,968.0	1,968.0	242s
60%	440	2,124.0	2,118.0	3600s	438	2,164.0	2,133.3	3600s
70%	495	2,334.0	2,334.0	1379s	492	2,290.0	2,256.0	3600s
80%	568	2,458.0	2,448.0	3600s	564	2,514.0	2,381.0	3600s
90%	599	2,640.0	2,640.0	159s	597	2,538.0	2,448.0	3600s
100%	708	2,874.0	2,736.0	3600s	706	2,868.0	2,723.5	3600s

Once the parameters had been defined, points were grouped in regions, based on accessibility data. Such data was not initially available and was determined using a robotic simulation software owned by the company. This software had 3D models for each vehicle and robot, allowing to verify before implementation if each of the welding points in a solution could, indeed, be reached. Each region was assigned accesses from which robots could reach it. Adjacency, macro-regions and cycle parameter tuples were also defined based on the observed geometrical characteristics. The instances' data were adapted in order to preserve both: the reproducibility and the company's private information.

The model and data from each product variant allowed the definition of Mixed Integer Linear Programming models for each vehicle. In order to evaluate the computational complexity in regard to the number of welding points. Instances were generated with various percentages of the total number of welding points. Each region's number of welding points was multiplied by a reducing factor and rounded down to an integer. These factors were chosen, in a manner that instances would have, approximately, 10%, 20%, ..., 90% and 100% of the total number of welding points. The 100% instances represent the practical cases. Data on the *Supporting Information* present the tuple and parameter sets required to define each problem instance. Model files (.lp format) are also provided, along with the best solution files for each instance, containing the value of the objective function and of each variable presented in the model file. Thus, reproducibility is assured by the provided detailed information.

4.2. Results

The computational testes were conducted using a Core i7 (2.3 GHz) computer with 8.0 GB of RAM and a universal MILP solver was employed to solve all instances. The computational time limit of each run was set to one hour. Table 4 illustrates the solution of the computational and practical case studies, each variant representing a different vehicle. The computational tests indicate that the number of points to be distributed influences the computational load. However, this number is not the only factor to be considered: Optimality was more difficult for Variant 4 problems than for Variant 1, despite the number of welding points. This means that other instance-specific factors such as accessibility and interferences might significantly affect the instance's computational difficulty.

For the practical cases, the company's mid-term productivity goal was a cycle time of 2880 T.U. One of the practical instances could be solved to optimality (relative gap = 0%), while the others couldn't (up to 5% relative gap). However, all of them allowed the cycle time to reach the company's goal (the value associated with the target productivity, measured in time units per vehicles). The model's gain means that the studied part of the manufacturing line has the capacity to produce around 6.6% more vehicles without employing a single new robot. The Figure 15 illustrates the station-wise gain in cycle time. The vehicle corresponding to the model that was solved to optimality was also the one that was produced the most, meaning that it tends to dictate the line's overall cycle time. Furthermore, according to Table 4, the obtained cycle time difference between models for the real-world case is smaller than 1% (2,868.0 *versus* 2,874.0) corroborating with the hypothesis that they could be treated separately rather than on a mixed-model (or multi-model) manner. All models did reach the main cycle time goal (2880 T.U.) and idle times in some of the robots can be used, for instance, to perform welding points that were not performed in this part of the manufacturing line.

The Figure 15 shows a comparison between the initial workload distribution between stations and the optimized configuration. By computing the trade-off between movements and number of performed points, the model is able to better distribute the workload and reduce the flow-shop's cycle time: In some cases, the sum of robot times (heavily linked to movements) increased by the optimization process, but the line's cycle time decreased. Notice that this has been achieved under the conservative parameter framework made clear by Figure 14 validation.

The obtained solutions to the practical instances were afterwards internally translated back from mathematical to tangible practical information and were verified to provide feasible solutions to the real-world problem by the company's specialists.

5. Conclusion

Balancing robotic spot welding manufacturing lines is a challenging problem. The time required to welding the points is short compared to cycle time. However, robots have to move between them

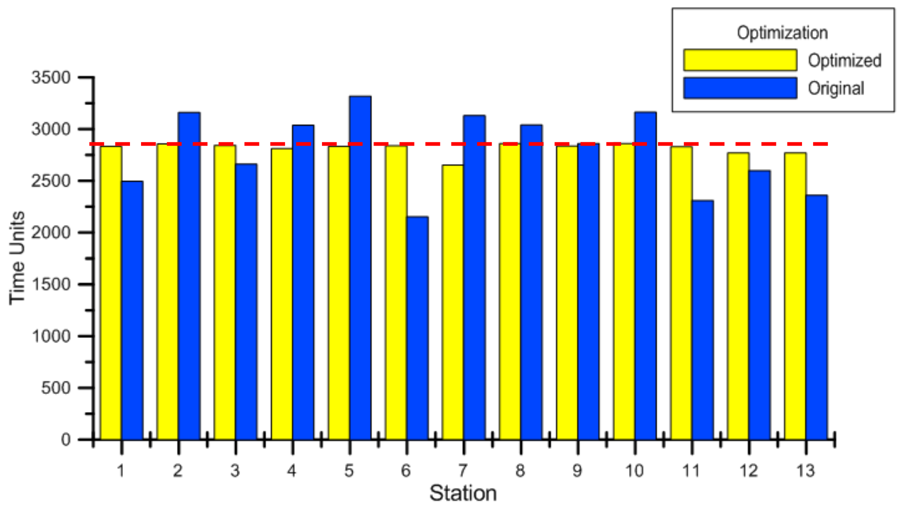

Figure 15: Comparison between the model's optimized solution for one of the instances and the empirically measured time: The dashed line highlights the smaller largest value for cycle time amongst stations.

and the movement time must be taken in account. Furthermore, geometrical aspects impose assignment constraints and multiple robots per station impose interference constraints. The combination of these restrictions lead to the understanding that previously described models could not properly describe the problem. A new (MILP) model is proposed and applied to solve a real-world scenario from a Renault factory in the outskirts of Curitiba, Brazil.

Practical case studies were conducted for four vehicle models on a large-size real-world line (42 robots and over 700 welding points for each model). Each of the models was optimized separately, and optimality was not reached for all instances. However, the obtained answers offered cycle time reductions of around 6.6%, allowing the company to reach its mid-term goal of productivity for that part of the factory without the need to buy new robots.

Computational studies have shown that the number of welding points is a relevant factor to the model tractability, but not the only one. Some small instances were more difficult than larger counter-parts (see Table 4). This indicates that instance-specific characteristics, such as accessibility and interference, are also relevant factors for computational tractability.

Instances were generated to describe each of the vehicle types produced in the factory. Parameters were set according to empirical observations. The line was a mixed-model one, however in this particular part of the factory all models could individually reach similar cycle times. This verified the hypothesis about the optimization process being able to set aside sequencing considerations and balance each model individually. The results obtained for the case studies were validated by the company specialists.

The new Mixed Integer Linear Programming model provides a framework to optimize robotic welding lines by offering a formulation to simultaneously deal with: Robot-wise variations on parameters, Assignment Restrictions, Movement Times and Interference Constraints. As future development, this basic framework can be adapted to incorporate other features such as feeder lines and parallel stations. Furthermore, in an assignment line design context, the presented model could be adapted to minimize the number of robots required or optimize a cost-oriented function.

Acknowledgments

We would like to acknowledge the financial support from Fundação Araucária (Agreement 141/2015) and CNPq (Grant 305405/2012-8). In addition, we would also like to show our appreciation to Renault for the opportunity of working with them and their manufacturing lines.

References

Andrés, C., Miralles, C., & Pastor, R. (2008). Balancing and scheduing tasks in assembly lines with sequence-dependent setup times. *European Journal of Operational Research*, *163*, 1212–1223.

Battaïa, O., & Dolgui, A. (2013). A taxonomy of line balancing problems and their solution approaches. *International Journal of Production Economics*, *142*, 259–277. doi:10.1016/j.ijpe.2012.10.020.

Bautista, J., & Pereira, J. (2007). Ant algorithms for a time and space constrained assembly line balancing problem. *European Journal of Operational Research*, *177*, 2016–2032.

Becker, C., & Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, *168*, 694–715.

Bowman, E. (1960). Assembly-line balancing by linear programming. *Operational Research*, *8*, 2016–2032.

Boysen, N., Fliedner, M., & Scholl, A. (2007). A classification of assembly line balancing problems. *European Journal of Operational Research*, *183*, 674–693.

Boysen, N., Fliedner, M., & Scholl, A. (2008). Assembly line balancing: Which model to use when? *International Journal of Production Economics*, *111*, 509–528.

Boysen, N., Fliedner, M., & Scholl, A. (2009). Sequencing mixed-model assembly lines: Survey, classification and model critique. *European Journal of Operational Research*, *192*, 349–373.

Buxey, G. M. (1974). Assembly line balancing with multiple stations. *Management Science*, *20*, 1010–1021.

Daoud, S., Chehade, H., Yalaoui, F., & Lionel, A. (2014). Solving a robotic assembly line balancing problem using efficient hybrid mehtods. *Journal of Heuristics*, *20*, 235–259.

Dar-El, E., & Rubinovitch, Y. (1979). MUST - A multiple solutions technique for balancing single model assembly lines. *Management Science*, *25*, 1105–1114.

Hillier, F. S., & Lieberman, G. J. (2015). *Introduction to Operations Research.* (10th ed.). Heidelberg: Mc Graw Hill.

Kim, H., & Park, S. (1995). Mixed model U-line balancing type-1 problem: A new approach. *International Journal of Production Research*, *33*, 2311–2323.

Lapierre, S. D., Ruiz, A. B., & Soriano, P. (2004). Balancing assembly lines: An industrial case study. *Journal of Operational Research Society*, *168*, 589–597.

Levetin, G., Rubinovitz, J., & Shnits, B. (2006). A genetic algorithm for robotic assembly line balancing. *European Journal of Operational Research*, *168*, 811–825.

Martino, L., & Pastor, R. (2010). Heuristic procedures for solving the general assembly line balancing problem with setups. *International Journal of Production Research*, *48*, 1787–1804.

Pastor, R., & Corominas, A. (2000). Assembly line balancing with incompatibilities and bounded workstation loads. *Ricerca Operativa*, *30*, 23–45.

Patterson, J. H., & Albracht, J. J. (1975). Assembly-Line Balancing: Zero-One Programming with Fibonacci Search. *Operations Research*, *23*, 166–172.

Pinto, P. A., Dannenbring, D. G., & M, K. B. (1983). Assembly line balancing with processing alternatives: an application. *Management Science*, *29*, 817–830.

Rubinovitz, J., & Bukchin, J. (1991). Design and balancing of robotic assembly lines. In *Proc. of the fourth world conference on robotics research*. Pittsburgh, PA.

Scholl, A. (1999). *Balancing and sequencing assembly lines.* (2nd ed.). Heidelberg: ed. Physica.

Scholl, A., Boysen, N., & Fliedner, M. (2008). The sequence-dependent assembly line balancing problem. *Operations Research Spectrum*, *30*, 579–609.

Scholl, A., Boysen, N., & Fliedner, M. (2013). The assembly line balancing and scheduling problem with sequence-dependent setup times: problem extension, model formulation and efficient heuristics. *OR Spectrum*, (pp. 291–320). doi:10.1007/s00291-011-0265-0.

Scholl, A., Fliedner, M., & Boysen, N. (2010). Absalom: Balancing assembly lines with assignment restrictions. *European Journal of Operational Research*, *200*, 688–701.

Thangavelu, S. R., & Shetty, C. M. (1971). Assembly line balancing by zero-one integer programming. *AIIE Transactions*, *3*, 61–68.

White, W. W. (1961). Comments on a Paper by Bowman. *Operations Research*, *9*, 274–276. doi:10.1287/opre.9.2.274.