A modified particle swarm optimization algorithm to solve the part feeding problem at assembly lines

Masood Fathi[a,b], Victoria Rodríguez[c], Dalila B. M. M. Fontes[d], Maria Jesus Alvarez[a]

[a] *Department of Industrial Organization, School of Engineering (Tecnun), University of Navarra, Pº Manuel Lardizabal 13, 20018 San Sebastian, Spain*

[b] *Laboratory of Artificial Intelligence and Decision Support (LIAAD), Institute for Systems and Computer Engineering of Porto (INESC Porto), Rua Dr. Roberto Frias, 4200-465 Porto, Portugal*

[c] *Economics and Management School, University of Navarra, Campus Universitario, 31080 Pamplona, Spain*

[d] *Faculdade de Economia da Universidade do Porto and LIAAD-INESC TEC, Rua Dr. Roberto Frias, 4200-464 Porto, Portugal*

Corresponding Author: Masood Fathi, Department of Industrial Organization, School of Engineering (Tecnun), University of Navarra, Pº Manuel Lardizabal 13, 20018 San Sebastian, Spain.
Laboratory of Artificial Intelligence and Decision Support (LIAAD), Institute for Systems and Computer Engineering of Porto (INESC Porto), Rua Dr. Roberto Frias, 4200-465 Porto, Portugal.
Tel. +34943219877, Fax. +34943311442, fathi.masood@gmail.com

A modified particle swarm optimization algorithm to solve the part feeding problem at assembly lines

Abstract

The Assembly Line Part Feeding Problem (ALPFP) is a complex combinatorial optimization problem concerned with the delivery of the required parts to the assembly workstations in the right quantities at the right time. Solving the ALPFP includes simultaneously solving two sub-problems, namely tour scheduling and tow-train loading. In this paper, we first define the problem and formulate it as a multi-objective mixed integer linear programming model. Then, we carry out a complexity analysis, proving the ALPFP to be NP-complete. A modified Particle Swarm Optimization (MPSO) algorithm incorporating mutation as part of the position updating scheme is subsequently proposed. The MPSO is capable of finding very good solutions with small time requirements. Computational results are reported, demonstrating the efficiency and effectiveness of the proposed MPSO.

Keywords: Part feeding problem, Assembly line, Tour scheduling, Tow-train loading, Particle swarm optimization.

1. Introduction

Manufactures must be able to efficiently deal with the large number of parts required to manufacture the large number of different products that their clients demand. Since most of these products have a common base that is then customized to satisfy customer needs and preferences, the so-called mixed-model assembly line is appropriate and frequently used. Relevant examples are provided by the automobile industry, where customers can choose from a wide range of individual options (e.g., sunroof, leather trim) resulting in a very large number of car models (see, e.g., Pil and Holweg, 2004).

Once parts have been received from suppliers, they must be delivered to the assembly line, either directly or after being temporarily stored. In the case of temporary storage, parts can either be stored at the central receiving store or in a decentralized logistics area, which has become known, at least in the automobile industry, as the Just-In-Time (JIT) supermarket (Battini et al., 2010). Decentralized storage allows for additional flexibility in part supply and reduces inventory costs (Battini et al., 2013). The parts are usually delivered to the assembly line from the supermarket through round trips by tow-trains.

Amongst the many problems related to part delivery in a supermarket concept, our focus is on the problem of deciding which parts, if any, should be delivered to the workstations on each tow-train tour and in what quantities. Several tours are scheduled to take place during a working day, and on each one a tow-train may serve several workstations. Each tour starts at the supermarket, where the tow-train is loaded and then goes around the shop floor, usually along a predefined route, delivering bins to their corresponding workstation. In general, this may involve three sub-problems, namely: routing, scheduling, and loading (Boysen et al., 2015). In the specific problem we address here, the tow-train route and the available tours are predefined, and thus we are only addressing the scheduling and loading sub-problems. Two decisions need to be made: more specifically we need to decide which of the available tours are taken and which parts, and in what amounts, are loaded on them. This problem was termed the Assembly Line Part Feeding Problem (ALPFP) in the literature (Fathi et al., 2014a). The objectives of ALPFP are defined as minimizing the number of tours taken, due to the costs associated with each tour, and minimizing part inventories at the workstations, because of limited space. The constraints considered are the tow-train capacity, the storage capacity at the workstations, and the part demand of each workstation. Here we address an extended version of ALPFP that we prove to be NP-complete, for which a Modified Particle Swarm Optimization (MPSO) algorithm is proposed.

The main contributions of this work are threefold. Firstly, an extension to ALPFP is presented based on experience acquired at a car manufacturing company. Secondly, we prove this new problem to be NP-complete. Finally, a meta-heuristic algorithm that is capable of finding good solutions is developed.

The remainder of the paper is organized as follows. Section 2 provides a literature review of related problems and previous solution methods. In Section 3, the problem is defined, a mathematical programming formulation is provided, and the problem is proven to be NP-complete. Section 4 discusses the solution procedure. In Section 5, we report on the computational experiments carried out and provide evidence of the method's efficiency and effectiveness. Finally, some conclusions are drawn and future research directions are pointed out in Section 6.

2. Literature review

Part logistics comprises a wide variety of decision problems. Since its main purpose is to ensure that assembly lines never stop due to an insufficient amount of parts, it involves both strategic and operational problems. For a recent and comprehensive discussion on part logistics problems and on the proposed methodologies, see the work by Boysen et al. (2015).

The work presented here addresses an ALPFP problem that is an extension of the one originally proposed by Fathi et al. (2014a). The original problem was solved using a simulated annealing algorithm and the results compared favourably to those obtained by CPLEX. A first extension to this problem was studied in Fathi et al. (2014b) by also considering a constraint on tour delivery time. A memetic ant colony-based heuristic algorithm (MACO) was proposed and the solutions obtained were compared with those of CPLEX.

These are the only works on this specific problem. Nevertheless, similar part delivery problems have been addressed before. Rao et al. (2013) addressed a part supply scheduling problem in which each tour visits only one workstation. Thus, for each tour the decisions to be made are which workstation to visit, the part quantity to be loaded and the tour depart time, such that travelling and inventory holding costs were minimized. Limits were imposed on tour capacity and on inventory at the workstations. The authors developed a backward-backtracking approach, a hybrid genetic algorithm, and a simulated annealing algorithm that take advantage of problem-specific properties.

Satoglu and Sahin (2013) tried to design an efficient JIT milk-run part supply system through simultaneously solving the routing and scheduling problems. The authors assumed that all the workstations were supplied via a single depot through a cyclic service. The problem was formulated as a non-linear mixed-integer programming (MIP) model, minimizing the total parts handled and the inventory costs, for which a heuristic called Route Construction Algorithm (RCA) was proposed.

Kilic and Durmusoglu (2013) also addressed the routing and scheduling problems. The authors tried to minimize work-in-process and transportation costs by designing efficient routes and finding the best time period. A linear MIP model was presented and it was assumed that part delivery was preformed through cyclic deliveries. A two-part heuristic method was proposed: the first part constructs routes and assigns workstations to them, while the second part attempts to minimize the number of tours by increasing the time between consecutive routes.

Emde et al. (2012) investigated the tow-train loading problem, which aimed to find the parts to be loaded on each tour, while avoiding material shortages given the limited capacity of the tow-trains. Tours and tour routes were given. Their objective was to minimize inventory near the line, which was translated into two objective functions, namely: minimize the total inventory at the workstations and minimize the maximum amount of inventory at a single workstation at any time.

Emde and Boysen (2012) addressed a part supply problem by sequentially solving two sub-problems: a partition problem, where workstations were divided among tow-trains, and a tour problem, where they determined the number of tours and respective starting times. Optimal solutions were obtained for each of these problems by using dynamic programming. The authors also studied the influence of cyclic and non-cyclic schedules on inventory. Their findings showed that non-cyclic schedules provided additional flexibility and allowed for a considerable reduction in inventories.

Golz et al. (2012) presented a case study in which they determined how to deliver parts to workstations in order to be able to meet a given assembly sequence. The approach decomposed the problem into two stages. In the first stage, transportation orders were derived from the assembly sequence and the bill of materials. In the second stage, a part supply problem had to be solved in order to decide which parts were to be delivered on each tour, taking into account capacity restrictions (tour and inventory at the workstations), part needs, and tour scheduling constraints.

Vaidyanathan et al. (1999) addressed the problem of planning vehicle routes to deliver parts in a JIT production plant. The problem was modelled by adding a non-linear capacity constraint to the standard vehicle routing problem, such that vehicle idle times and inventories at customer locations were minimized. The authors proposed a heuristic procedure comprising two phases. In the first phase, a nearest neighboured algorithm was used to find capacity-feasible routes, while the second phase improved these routes by using a 3-opt heuristic.

The literature review shows that the assembly line part feeding problem (ALPFP) is a very relevant problem in industry, particularly in the automobile industry, and it has been increasingly gaining attention from academics. Most of the works reported are based on a specific real world problem. As a result, the problems addressed, although similar, include different constraints and/or objectives. In order to better compare the current problem and the works reviewed here, in Table 1 we provide a summary of the problem features and solution methods proposed.

It should be noted that our work includes a new constraint on the maximum weight that can be loaded on the tow-train. This constraint, despite its importance due to the difference in tow-train type (e.g. light, heavy), the power of the towing vehicle, and safety policies, has not been considered before. We address this problem version by proposing a novel particle swarm optimization algorithm.

[Insert Table 1 Here]

3. Problem description and model formulation

In the ALPFP addressed in this work there are N possible tow-train tours out of which we must choose the ones that will be used. A tow-train tour starts at the single supermarket, where it is loaded with full bins, then goes around the shop floor, passing every workstation using the predetermined path to deliver the bins to the respective workstations and collect empty bins, and ends back at the supermarket. Thus, for each tour taken we must also determine how many bins of each part type will be loaded. Tour limits exist regarding the number and weight of the loaded bins and replenishment time. As parts are delivered using full bins only, bins are used for one part type at a time, and bins are collected only if empty; inventory at the workstations may include full and partially empty bins. Since the space available at the workstations is scarce, there is a limit on the inventory at the workstations.

Our objectives are to minimize the number of tours taken (primary objective) and to minimize the sum of the part inventories at the workstations (secondary objective). These objectives, which are dealt with using a lexicographic ordering, reflect the need to reduce the costs incurred with the tours and to reduce storage at the workstations, which is in line with JIT philosophy. Part needs are known and must be met while respecting tour, tow-train[1], and storage space capacities.

The next sections present the mathematical programming model and the proof of the problem complexity.

[1] Tow-train capacity is transformed into tour capacity by adding the capacity of all available tow-trains. Thus, here and hereafter, when referring to capacity limits, tour and tow-train will be used interchangeably.

3.1 Model formulation

The model presented here is a modified and extend version of the model proposed by Fathi et al. (2014b). The notation used in the formulation of the ALPFP is summarized below, and it is followed by the model and an explanation.

Sets:

I: set of parts;

T: set of tours;

Indices:

i: part index, $i \in I$;

t: tour index, $t \in T$;

Parameters:

M : number of parts, $M = |I|$;

N : number of tours, $N = |T|$;

TB : tour bin capacity (in bins);

TW : tour weight capacity (in kilograms);

TT : tour replenishment time limit (in seconds);

SC_i: storage capacity at the workstation requiring part *i* (in bins);

w_{it}: weight of a bin filled with part *i* (in kilograms) in tour *t*;

rt : bin replenishment time, loading and unloading (in seconds per bin);

d_{it}: demand for part *i* in tour *t* (in bin fractions);

IL_{i0}: initial inventory for part *i* at the beginning of the first tour.

Decision variables:

b_{it}: bins of part *i* delivered in tour *t*;

$$Y_t = \begin{cases} 1, & \text{if tour t is taken,} \\ 0, & \text{otherwise;} \end{cases}$$

IL_{it} : inventory at the workstation processing part *i* after delivery of tour *t*.

Model:

$$minimize \left\{ \sum_{t=1}^{N} Y_t ; \sum_{i=1}^{M} \sum_{t=1}^{N} IL_{it} \right\}, \qquad (1)$$

Subject to:

$$b_{it} + IL_{it-1} - d_{it} = IL_{it}, \qquad \forall\ i \in I\ and\ \forall\ t \in T, \qquad (2)$$

$$\sum_{i=1}^{M} b_{it} \le \varphi Y_t, \qquad \forall\ \mathrm{t} \in \mathrm{T}, \qquad (3)$$

$$b_{it} + IL_{it-1} \le SC_i, \qquad \forall\ i \in I\ and\ \forall\ t \in T, \qquad (4)$$

$$\sum_{i=1}^{M} b_{it} \times w_i \le TW, \qquad \forall\ \mathrm{t} \in \mathrm{T}, \qquad (5)$$

$$IL_{it} \ge 0, \qquad \forall i \in I\ and\ \forall t \in T, \qquad (6)$$

$$b_{it} \ge 0\ and\ integer, \qquad \forall i \in I\ and\ \forall t \in T, \qquad (7)$$

$$Y_t \in \{0,1\}, \qquad \forall\ \mathrm{t} \in \mathrm{T}. \qquad (8)$$

The objective function, given by equation (1), includes two objectives and thus the problem is formulated as a multi-objective mixed linear programming model. This type of models may be studied from different viewpoints, discussion on them is provided in the comprehensive survey by Coello Coello (1999). In this work, we use lexicographic ordering, in which the minimization of the number of tours taken is the primary (most important) objective and the minimization of the total inventory at the workstation is the secondary objective. To solve it, we first obtain solutions satisfying the problem constraints that optimize the primary objective. Then, the optimization of the secondary objective is constrained to the optimal solution space of the first objective, which is done by adding a constraint. (In our problem this constraint limits the available tours to the ones chosen while optimizing the first objective.) Thus, the secondary objective will be optimized in a meaningful manner if its value varies within this solution space.

Equation (2) represents the balance constraints, which in this case impose that part demands be satisfied. Equation (3) ensures not only that bins are only delivered through tours taken but also that tour capacity both in terms of number of bins and replenishment time are satisfied. Note that φ represents the largest possible number of bins that simultaneously ensures tour bin capacity and tour replenishment time capacity, and it is given in equation (9). Equations (4) and (5) enforce storage capacity at the workstations and tour weight capacity, respectively. Finally, equations (6) to (8) state the nonnegative, integer, and binary nature of the decision variables.

$$\varphi = min\left\{ \left\lfloor \frac{TT}{rt} \right\rfloor, TB \right\}. \tag{9}$$

3.2 Complexity proof

In this section, we show that even finding a feasible solution for the ALPFP is strongly NP-complete via a transformation from the 3-partition problem, which is well known to be strongly NP-complete (Garey and Johnson, 1979).

3.2.1 3-partition problem

In the 3-partition problem we are given a set Z = $\{1, 2, ..., 3q\}$ of 3q positive integers

a_m (m=1,…,3q) and a positive integer G such that G/4<a_m<G/2 and $\sum_{m=1}^{3q} a_m = qG$. The problem is to determine whether a partition of set Z into q subsets $\{H_1, H_2, ..., H_q\}$, each containing exactly 3 elements from Z, exists such that $\sum_{m \in H_u} a_m = G, \forall u = 1, ..., q$.

3.2.2 Transformation from the 3-partition problem into ALPFP

Consider 3q bins of different parts, each with a specific weight $w_m = a_m, \forall m = 1, ..., 3q$. These bins, which have a total weight of qG, must be delivered to the appropriate workstations through the use of N = q tours. Each tour has a weight capacity limit of TW= G. Moreover, let the total demand of each part at the final period be $d_{m,N} = 1, \forall m = 1, ..., 3q$, and zero in all other periods. Further, assume that there are no limits on the number of bins that can be delivered on each tour φ and no limits on the storage capacity at the workstation SC_m. The question we ask is whether a feasible solution for the ALPFP without part shortages can be found.

First, a feasible solution for an instance of a 3-partition can be directly transformed into a feasible solution to the above stated ALPFP instance. Let each subset H_u be assigned to a single tour. This means that q tours are used to deliver the required parts. Since the sum of the integers in each subset, i.e., tour amounts to G, the tour weight limit is satisfied. Therefore a feasible solution to the ALPFP has been obtained.

In addition, we can also prove that each feasible solution to the ALPFP is also a YES for the 3-partition problem. This will be shown by proving that every single tour must have exactly 3 bins. On the one hand, any tour containing more than 3 bins leads to tour overload (excessive weight), since bins weights must satisfy $a_m > G/4$.

On the other hand, if a tour T_a with fewer than three bins exists, there has to be another tour T_e with more than three bins, which would inevitably cause weight overload on tour T_e. Thus, any feasible solution for ALPFP must contain exactly three bins per tour. In addition, if there exists a tour that does not fully use tour capacity in terms of weight, even if with three bins, overload will occur on another tour, because its weight would exceed the limit.

It directly follows that a feasible solution to the ALPFP exists if and only if the answer to the corresponding instance of the 3-partition problem is YES. This proves the NP-completeness, in the strong sense, of finding a feasible solution to the ALPFP.

4. The proposed PSO algorithm

We propose to find solutions to the ALPFP by resorting to Particle Swarm Optimization. Given the NP-completeness of the problem, only heuristic approaches are capable of finding good solutions in a reasonable amount of time, at least for large problem instances.

Particle swarm optimization (PSO) is a population-based stochastic optimization technique developed by Kennedy and Eberhart (1995), who were inspired by social group behaviour of bird flocking or fish schooling. The system is initialized with a population of random solutions, termed particles, and searches for an optimum solution by updating the particles' position and velocity according to their own experience and the experience of their neighbours. PSO usually has no evolution operators such as crossover and mutation, but this is one of the features we introduce in the MPSO proposed here since we use mutation operators when updating the position vector.

In the following subsections, we provide details about the modified version of a discrete PSO proposed here to address the ALPFP. In particular, we discuss how particles are represented and how they move around the search-space, i.e. how the position and velocity vectors are represented and updated. Then, we show how feasible solutions to the ALPFP are obtained and evaluated.

4.1 Solution approach

We propose a new approach based on a modified PSO algorithm, MPSO. The MPSO main features are: (i) it uses an indirect representation base on permutations of a set of priority rules and a 2-phase hierarchical constructive procedure that uses the aforementioned rules to obtain feasible solutions to the ALPFP; (ii) particles' positions are updated by using two mutation operators, namely insert and swap. Solution quality is evaluated, as usual, through the use of a fitness function, which is obtained by combining the two defined objective functions.

The role of the MPSO is to evolve the encoded parameters which are the input to the solution constructor. This is achieved by updating the velocity vector and the position vector (the latter by using mutation operators).

For each particle, the following phases are applied:

1. Solution construction 1: The first phase finds a feasible solution (the tours to be taken and the bins loaded on each tour), optimizing the primary objective, i.e. minimizing the number of tours taken.

2. Solution construction 2: The second phase seeks a better feasible solution regarding the secondary objective, i.e. the minimization of the total inventory, within the optimal solution space of the primary objective. Thus, the secondary objective will only be optimized in a meaningful manner if its value varies within this solution

space.

3. Fitness evaluation. This phase computes the combined value of the defined objective functions, as given in equation (12).

4.2 Particle position and velocity

One of the key issues in designing a successful PSO algorithm is the representation step, which aims to find an appropriate mapping between the problem solution and the PSO particle. A solution is represented as a vector of natural numbers in the range $[1, m]$, each representing a priority rule of which there are m. These rules are to be used in the solution construction procedure to select a part number to assign to each tow train on each tour and are provided in Table 2. The length of the vector $2n$ is the double of the maximum number of bins that may be assigned at any step. The value of n is found by computing the difference between the total demand in bins and the number of bins that must be delivered on the first tour (although it may include others), as in equation (10). The first n values are used in the first part of the hierarchical construction procedure, while the last n values are used in the second part.

$$n = \sum_{i=1}^{M}\sum_{t=1}^{N} d_{it} - \sum_{i=1}^{M} d_{i1} - IL_{i0}. \tag{10}$$

The novelty of our position representation allows the best priority rule permutation to be obtained rather than the best particle positions. Since the permutations are used to construct solutions, each solution is associated with a single permutation of priority rules. Thus, finding a good solution corresponds to finding a good permutation of priority rules.[2]

Velocity also plays a significant role in guiding the particles in adjusting their distance and moving towards better positions (solutions). In our algorithm, the velocity of each particle is a vector that has the same length as the position vector, with real numbers in the range $\left[-V_{max}, V_{max}\right]$.

[Insert Table 2 Here]

4.3 Particle updating

Particles adjust their positions and velocities according to a "Psychosocial compromise'' between what an individual is comfortable with and what society reckons. The particle position is probabilistically updated by a neighbourhood-based mutation, which uses the velocity to find out which heuristic the mutation is applied to. In our case, a vector element with higher velocity means that the heuristic associated with it (identified by the corresponding element of the position vector) may not be placed in a good position within the sequence of heuristics. Thus, the permutation should be changed by removing it from the current position. This can be done in two ways: removing the priority rule, which is then replaced by a new one, or swapping this priority rule with some other. A specific element has a nonzero probability of not being changed; however if it is, then a mutation operator is applied.

The velocity vector is updated as in equation (11), where C_1 and C_2 are positive constants representing cognitive and social learning coefficients,

[2] Hereafter, we will use priority rules or heuristic interchangeably.

respectively, and V_{pk} and X_{pk} are the velocity and position of particle p at iteration k.

$$V_{pk+1} = W_k \times V_{pk} + C_1 \times R_1 \times \left(P_{best,p} - X_{pk}\right) + C_2 \times R_2 \times \left(G_{best} - X_{pk}\right). \qquad (11)$$

$P_{best,p}$ represents the best position that the pth particle has visited since the first time step, i.e., the permutation of priority rules corresponding to the best parts delivery solution ever found by particle p, and G_{best} represents the best position that any particle has visited from the beginning of the algorithm. At each iteration, these values, when bettered, are updated. R_1 and R_2 are two random numbers uniformly drawn from $[0,1]$ and W_k is the inertia weight factor for iteration k, which is used to achieve a good balance between exploitation and exploration. Larger values lead to exploration of the search space, thus preventing local optimum entrapment, while smaller values result in exploitation, thus intensifying the search around the current solution, taking advantage of the iterative process that has led us there. Usually, W values decrease throughout the search process, and we compute them as $W_{k+1} = \theta \times W_k$.

The updated position vector is obtained by considering each selected element of the vector for mutation. The swap and insert mutation operators are applied to a proportion of elements given by δ_1 and δ_2, respectively, where $\delta_1 + \delta_2 < 1$. Therefore, some of the position vector elements remain unchanged. Both mutation operators update the position vector elements by using information from the personal best position vector (P_{best}) with probability α, and from the algorithm best position vector (G_{best}) with probability $1-\alpha$, where $\alpha < 1$. A detailed description of how the position is updated is provided in Appendix 1.

4.4 Fitness

As mentioned above, we have defined two objectives for the ALPFP. Therefore, once feasible solutions have been obtained, they must be evaluated, not only to find out how good each solution is, but also to direct the search towards better solutions, i.e., guide the particles towards good positions. In order to do so, we have defined a fitness function that combines the two objective functions through the Minimum Deviation Method (Özcan and Toklu, 2009), as given in equation (12).

$$Obj.F = \beta_1 \left(\frac{f_1 - f_1^{\min}}{f_1^{\max} - f_1^{\min}} \right) + \beta_2 \left(\frac{f_2 - f_2^{\min}}{f_2^{\max} - f_2^{\min}} \right), \tag{12}$$

where $f_1^{\min}$ and $f_2^{\min}$ are the best values found so far for the first and second objectives; $f_1^{\max}$ and $f_2^{\max}$ are the worst values and f_1 and f_2 are the current values. β_1 and β_2 are coefficients associated with the first and second objectives, and they are used to establish the relative importance of the defined objectives. In this work, based on a real case study conducted at the Volkswagen (VW) assembly plant in Navarra (Spain), a very strong priority is given to the first objective (minimize the number of tours), thus $\beta_1 >> \beta_2$.

4.5 Hierarchical construction procedure

We adopt a hierarchical strategy to find feasible solutions. First, we find the sequence of bins to be delivered. This is done while aiming to use the smallest number of tours, which is the primary objective (tour-oriented procedure). This problem, however, might have several possible such solutions. If that is the case, then we are interested in finding one that finds the sequence that minimizes the sum of the inventory at the workstations, which is the secondary objective (part-oriented procedure).

Tour-oriented procedure: starting from the first tour, parts and part bins are selected and assigned to the tour until one of the tour limits (B-number of bins, W-weight, or A-time) or the capacity at the workstations is reached. Then the procedure is repeated by considering the next tour, say tour *t*, if and only if the parts inventory at the workstations is not enough to satisfy the part needs in tour *t*, for at least one part. Otherwise, the current tour will not be performed, i.e. it will be removed, and the assigning process is resumed by considering the first tour L ($L \geq t+1$) for which at least one part demand cannot be satisfied from the inventory at the workstations. This iterative process is continued until all required bins have been assigned to tours. Bins are assigned to each tour to satisfy demand (direct assignment) and to fully load the tour, if there is additional capacity available (indirect assignment). The choice of which bins to assign to each tour for indirect assignment is made by using one of several possible heuristic rules. The rule used is determined by the value of the current element of the position vector. The main output of this phase is the tours taken, whose number is minimized. A detailed description is provided in Figure 1.

[Insert Figure 1 Here]

Part-oriented procedure: a part number is selected at the time. For the part under consideration we assign bins, in a JIT fashion, to each of the tours selected in the first part. That is, bins are loaded on tours in the smallest possible amount that satisfies demand and as close as possible to when it is needed. This way, the total inventory at the workstations is minimized. However, such a procedure may possibly result in an unfeasible solution since the last tour before each removed tour (or the last tour taken if none has been removed) may have one or more of its limits exceeded (otherwise demand would not be satisfied). Thus, a backward recursive procedure is used to check tour feasibility and to restore it whenever any of the tour limits are exceed. This done by shifting bins backward until a feasible solution is obtained. Parts to be moved backward are chosen according to the priority rules indicated by the second part of the position vector. A detailed description is provided in Figure 2.

[Insert Figure 2 Here]

A detailed example illustrating the hierarchical constructing procedure is provided in Appendix 2.

5. Computational experiments

To assess the performance of the proposed MPSO, we randomly generated 12 problem instances with varying characteristics, which can be downloaded from http://www.tecnun.es/documents/10229/42909/PSO_Instances.xlsx/e098d878-8758-46fb-a920-1e1d6d45331a. Three problem sizes have been considered: small (fewer than 50 parts), medium (between 50 and 100 parts), and large (more than 100 parts). For each of these instances, four different values for the number of available tours N have been considered. In addition, we have also solved a case study from VW-Navarra. Therefore, in total we solved 49 problem instances. The case study was also used to help to define values and value ranges for the parameters of the problems we generated.

The part demands, weights, initial inventory, and storage capacity at the workstations are random numbers uniformly distributed as follows: d_{it} in [5, 100]; w_i in [5, 20]; IL_{i0} in $\left[0, SC_i\right]$; SC_i in [λ_i, 10], where λ_i is twice the average tour delivery for part i using all available tours (N). Based on the case study, we assume that no more than 100 bins can be loaded on each tow-train, since tow-trains cannot have more than two wagons and each wagon cannot take more than 50 bins. For the same reason, we also assume that replenishment time is 25 seconds per bin and the planning horizon is 86400 seconds (one day).

For comparison purposes, we have also solved the problems using both the heuristics proposed (cf. Table 2) and CPLEX 12.4. The MPSO and the heuristics were coded in MATLAB 2010b. All solution methodologies were run on a personal computer with a 3.3 GHz Intel Core i3 CPU and 8 GB of RAM. Moreover, CPLEX was run with a time limit (3600 seconds).

The parameters required by the MPSO were set after we conducted an analysis of their influence on the performance of the methodology. In order to do this, we used recommended values from the literature as a starting point and then resorted to the Taguchi method (Taguchi et al., 2005) to reduce the number of combinations and provide maximum coverage with a minimum number of test cases. Two main criteria were used to select the final parameter values: maximizing signal to noise ratio and minimizing the mean response (minimization problem). The number of iterations (swarms) was set to 50 and the number of particles to half of the number of parts considered ($[M/2]^+$). The other parameters were set as follows: $V_{\max} = [M/2]^+$, $W = 0.9$, $\theta = 0.98$, $\delta_1 = \delta_2 = 0.1$, $\alpha = 0.5$, $C_1 = C_2 = 0.5$.

Table 3 reports the computational results for the MPSO, the CPLEX and the heuristic (best solution obtained from among the 10 priority rules). For each problem instance we give a summary of its characteristics (*M*–number of parts; *N*–number of available tours; *TB*–tour bin capacity; *TW*–tour weight capacity; *TT*–tour time capacity) and solution performance (T^* –number of used tours; *AIL*–average part inventory; *MAD*–smooth coefficient across tours; CPU(s)–CUP time in seconds).

The *AIL* and the *MAD*, which allows for comparison of workload variation across tours (Rachamadugu and Talbot, 1991), are computed as in equations (13) and (14), respectively.

$$AIL = \left(\sum_{t=1}^{N} \sum_{i=1}^{M} IL_{it} \right) \Big/ \left(N \times M \right). \tag{13}$$

$$MAD = \left(\sum_{t=1}^{N} \left| \sum_{i=1}^{M} b_{it} - \tau \right| \right) \Big/ T^*, \text{ where } \tau \text{ is calculated as } \sum_{i=1}^{M} \sum_{t=1}^{N} d_{it} \Big/ T^*. \tag{14}$$

[Insert Table 3 Here]

Table 3 shows both the quality of the solutions obtained (*T^*, AIL, MAD*) and the computational efficiency (CPU (s)). The MPSO obtained a better solution with regard to the primary objective (number of tours) than the CLPEX for four problem instances and for seven problem instances if compared with the best solution provided by the 10 heuristics. Even when compared with the best solution found by any of the alternative methods (CPLEX and heuristics) the MPSO always found as good a solution or better (marked with "*"). In terms of the secondary objective

(*AIL*), the solutions can only be compared when the number of tours found is the same. The CPLEX was able to find better values for seven problem instances (marked in bold). With regard to MAD, the proposed MPSO found the same result as CPLEX or better in most of the cases (6 better, 3 worse and 40 the same).

The heuristic solutions always imply a substantially larger average inventory, even when the number of tours is larger. Therefore, one would expect these solutions to have a smoother workload. However, this is not the case since for several problem instances the solutions provided by the heuristics imply a larger MAD, in addition to requiring a larger number of tours and a larger average inventory.

In terms of computational time, the results show that CPLEX was not able to obtain an optimal solution for over 50% of the problem instances considered (25 out of 49), due to the computational time limit set (3600 seconds). Regarding the MPSO, it was able to find very good solutions (always as good as or better than the alternative methods) within a very short computational time (averaging 12.39 seconds per instance). It is also worth noting that the CPU time for the best reported results achieved by the priority heuristics for all instances was less than 1 second.

Finally, we would like to point out that the solution obtained for the real case analysed in this study is currently being used in VW-Navarra.

6. Conclusions

In this paper, a real world ALPFP problem was addressed. The problem was an extension of Fathi et al. (2014b), which is based on the problem faced by VW-Navarra. The ALPFP consists of two sub-problems, namely tour scheduling and tow-train loading problems. We developed a mixed integer linear programming model and proved the problem to be NP-complete. Therefore, we proposed a modified PSO algorithm that is capable of finding very good solutions with small computational requirements. The quality of the solutions found was evaluated by considering the number of tours required to supply the workstations as a primary objective and the total inventory at the workstations as a secondary objective. The efficiency and effectiveness of the MPSO was demonstrated by solving a set of 48 randomly generated problem instances and a real one provided by VW-Navarra. All problem instances were also solved by CPLEX and by a set of 10 heuristics that are also proposed here. When compared with CPLEX, the MPSO, in addition to being much faster, found solutions which are equally good or better (for two problem instances) for the primary objective. Regarding the secondary objective, CPLEX was able to find a better solution for seven problem instances. Moreover, the best solution provided by the 10 heuristics considered was never better than those of the MPSO, regardless of the objective considered. Despite that, the proposed heuristics have the advantage of being extremely fast; the computational time required was always under 1 second.

Future work may include additional problem features and thus generalizing the problem addressed. Amongst the interesting and realistic features we have identified the following: non-identical bins, since a wide diversity of parts is common; non-identical tow-trains, due to the possibility of using one or more wagons and to the existence of different wagons; and delivering the same part to different workstations, although some parts are only used in a specific workstation while others, perhaps due to a more supporting role, may be needed in several workstations. In the problem addressed we consider that tour routes and possible schedules were given. However, we may also consider including such decisions, simultaneously addressing the routing problem, since regarding routes the when (departure time) and how (specific route) is dependent on where the parts are delivered and how many part bins are delivered. Furthermore, routes, which may have to be identical or be allowed to vary from tour to tour may be constrained, or optimized, in terms of the number of workstations visited and the distance travelled on each tour, the number of wagons used, and the tour frequency, amongst other constraints.

Acknowledgments

The authors greatly appreciate the collaboration of Volkswagen Navarra, the department of logistics in particular, as well as the Volkswagen Chair Universidad de Navarra for partially funding this research. This work was also partially funded by the Basque government under project named ‘Saiotek: MEPLAM’, and Energy-Aware Feeding Systems Project (EASY), an ERA-NET project of the ECO-INNOVERA 1st Joint Call, as well as the project "NORTE-07-0124-FEDER-000057", funded by the North Portugal Regional Operational Programme (ON.2 - O Novo Norte), under the National Strategic Reference Framework (NSRF), through the European Regional Development Fund (ERDF); by ERDF (FEDER) through the COMPETE Programme, and by national funds, through the Portuguese funding agency, Fundação para a Ciência e a Tecnologia (FCT) within project FCOMP-01-0124-FEDER-037281. The authors would like to thank Prof. Dr. Nils Boysen from Friedrich Schiller University Jena for his valuable comments and suggestions to improve the quality of the paper.

References

Battini, D., Boysen, N., Emde, S., 2013. Just-in-Time supermarkets for part supply in the automobile industry. J Manage Cont 24(2), 209–217.

Battini, D., Faccio M., Persona, A., Sgarbossa, F., 2010. “Supermarket warehouses”: stocking policies optimization in an assembly-to-order environment. Int J Adv Manuf Technol 50 (5-8), 775–788.

Boysen, N., Emde S., Hoeck M., Kauderer M., 2015. Part logistics in the automotive industry: Decision problems, literature review and research agenda. Eur J Oper Res 242 (1), 107-120.

Coello Coello, C. A., 1999. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. Knowl Inf Syst 1(3), 269-308.

Emde, S., Boysen, N., 2012. Optimally routing and scheduling tow trains for JIT-supply of mixed-model assembly lines. Eur J Oper Res 217 (2), 287-299.

Emde, S., Fliedner, M., Boysen, N., 2012. Optimally loading tow trains for JIT-supply of mixed-model assembly lines. IIE Trans 44 (2), 121-135.

Fathi, M., Alvarez, M.J., Mehraban, F.H., Rodríguez, V., 2014a. A Multiobjective Optimization Algorithm to Solve the Part Feeding Problem in Mixed-Model Assembly Lines. Math Probl Eng 2014 (1), 1-12.

Fathi, M., Alvarez, M.J., Rodríguez, V., 2014b. A novel memetic ant colony optimization-based heuristic algorithm for solving the assembly line part feeding problem. Int J Adv Manuf Technol 75 (1-4), 629-643.

Garey, M.R., Johnson, D.S., 1979. Computers and intractability: a guide to the theory of NP-completeness. 1st ed. San Francisco, CA: Freeman.

Golz, J., Gujjula, R., Günther, H.O., Rinderer, S., Ziegler, M., 2012. Part feeding at high-variant mixed-model assembly lines. Flex Serv Manuf J 24 (2), 119-141.

Kennedy, J., Eberhart, R., 1995. Particle Swarm Optimization. Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth, Western Australia (pp. 1942–1948).

Kilic, H.S., Durmusoglu, M.B., 2013. A mathematical model and a heuristic approach for periodic material delivery in lean production environment. Int J Adv Manuf Technol 69 (5-8), 977-992.

Özcan, U., Toklu, B., 2009. A new hybrid improvement heuristic approach to simple straight and U-type assembly line balancing problems. J Intell Manuf 20(1), 123-136.

Pil, F.K., Holweg, M., 2004. Linking product variety to order fulfilment strategies. Interfaces 34 (5), 394–403.

Rachamadugu, R., Talbot, B. (1991). Improving the equality of workload assignments in assembly lines. Int J Prod Res 29(3), 619–633.

Rao, Y.Q., Wang, M.C., Wang, K.P., Wu, T.M., 2013. Scheduling a single vehicle in the just-in-time part supply for a mixed-model assembly line. Comput Oper Res 40(11), 2599-2610.

Satoglu, S.I., Sahin, I.E., 2013. Design of a just-in-time periodic material supply system for the assembly lines and an application in electronics industry. Int J Adv Manuf Technol 65 (1-4), 319-332.

Taguchi, G., Chowdhury, S., and Wu, Y., 2005. Taguchi quality engineering handbook. New York: John Wiley and Sons.

Vaidyanathan, B., Matson, J., Miller, D., Matson, J., 1999. A capacitated vehicle routing problem for just-in-time delivery. IIE Trans 31(11), 1083-1092.

Appendix 1.

To better understand the position updated process a detailed description is provided in Figure A1.1.

[Insert Figure A1.1 Here]

Appendix 2. An Example

Let us consider an example that includes 5 part references, and a total demand of 24 bins. Table A1 provides the details on part demands (total and per tour) and weight, initial inventory, and capacity at each workstation. The tour capacity limits are 8 bins and 19 kilograms. Bin replenishment time is 25 seconds per bin and the total delivery time per tour cannot exceed 175 seconds. The maximum number of tours is 5.

[Insert Table A2.1 Here]

As explained before, the solution construction procedure makes use of the particle position to select the next part, whenever a choice exists. In this case, since the total demand is 24 and in the first tour we must deliver at least 5 bins, the dimension of the position vector is $38\left(2\times(24-5)\right)$. The first part (elements 1 to 19) is used in the tour-oriented procedure, while the second part (elements 20 to 38) is used in the part-oriented procedure.

Solution construction Phase 1 (tour scheduling)

Given the initial inventory IL_0=(0,0.5,1,0,0), assign to the first tour the per tour part demands deducted of the initial inventory. Let this be represented by b_1=(1,1,0,2,1). Thus, B_1=5 bins are assigned with a total weight of W_1=17kg and take A_1=125s to be delivered. Since none of the tour limits is reached, more bins can be delivered on this tour. Due to the weight restriction, the parts that have the possibility of being selected are parts 1 and 2. The selection is then made by using the heuristic determined by the first element of the position vector. Assume the first part of the position vector to be X=(1, 7, 5, 2, 8, 2, 1, 3, 5, 4, 7, 6, 1, 9, 5, 2, 9, 5, 2). Thus, in this case we use heuristic 1, which corresponds to selecting the part having the maximum demand, i.e., part 2. Thus, the load of tour 1 becomes b_1=(1,2,0,2,1). Since the weight limit has been reached, the tour is complete. Given the tour demand we are left with the inventory at the workstations IL_1=(0,1.3,0.4,0.8,0.2), which means that at the workstations we have S_1=(0,2,1,1,1) bins.

A second tour is then started. Given the per tour demand and the current inventory IL_1=(0,1.3,0.4,0.8,0.2), tour 2 has to deliver at least one bin of parts 1, 3, 4, and 5, i.e., b_2=(1,0,1,1,1). Thus, this tour comprises B_2=4 bins weighing W_2=14kg and it takes A_2=100s to deliver them. Since at the workstations there are S_2=(1,2,2,2,2) bins, more bins can be delivered on this tour. The parts available for selection are 1, 2, and 4, since workstation capacity for parts 3 and 5 has been reached. The second element of the position vector is 7, thus part 4 is selected since it is the one with the largest weight. The tour load is now b_2=(1,0,1,2,1) and the corresponding inventory is IL_2=(1,1.3,1.4,2.8,1.2). Since the tour total weight has been reached, the tour is complete.

Given the inventory left from the previous tour IL_2=(0,0.1,0.8,1.6,0.4), the third tour is started with b_3=(1,2,0,0,1) and thus S_3=(1,3,1,2,2). Since there is still available capacity within tour 3 and not all workstations have the inventory at capacity, we repeat the procedure explained above until the final tour is reached, in this case with b_3=(3,2,1,0,1), S_3=(3,3,2,2,2), B_3=7, W_3=17kg., and A3=175s.

A fourth tour is started. Given the inventory IL_3= (2,0.9,1.2,0.4,0.6) all undelivered demand can be fit on this tour without violating any limits. The solution obtained, see Table A2, is one that optimizes the primary objective, i.e. minimum number of tours taken.

[Insert Table A2.2 Here]

Solution construction Phase 2 (tour loading)

As explained previously, in this phase (which has 2 steps) we look for a solution that minimizes the total inventory at the workstations (secondary objective), without increasing the number of tours taken found in the previous phase (primary objective). In order to do so, we postpone parts delivery to the latest possible tour amongst the ones found in the previous phase. Thus, to each tour, except the last, and for each part we assign the minimum possible number of bins that ensures demand satisfaction. Then, the remaining demand will be assigned to the last tour.

Given the initial inventory $IL_0=(0,0.5,1,0,0)$, in tour 1 we assign 1 bin of parts 1, 2, and 5 (given the initial inventory their need is 1 or fewer) and 2 bins for part 4 (since its initial inventory is zero and the demand is larger than 1 and smaller than 2). No bin is assigned for part 3 as the initial inventory covers its demand. Thus, $b_1=(1,1,0,2,1)$, the inventory is $IL_1=(0,0.3,0.4,0.8,0.2)$ and the number of bins is $S_1=(0,1,1,1,1)$. Tour 2 is loaded with $b_2=(1,1,1,1,1)$, since this is the smallest amount needed to satisfy tour demand. Thus, the inventory and number of bins left are $IL_2=(0,0.1,0.8,0.6,0.4)$ and $S_2=(0,1,1,1,1)$, respectively. Similarly, tour 3 is loaded with $b_3=(1,2,0,1,1)$, providing $IL_3=(0,0.9,0.2,0.4,0.6)$ and $S_3=(0,1,1,1,1)$. Finally, we assign the remaining bins to the last tour (tour 4), in this case $b_4=(2,2,1,2,1)$.

The last tour is loaded with 8 bins weighing 25kg and it takes 200s to deliver them. This solution is clearly not feasible since on tour 4 both the weight and time limits are exceeded. The second step of the tour loading procedure is then initiated and it pushes back some bins in order to reach feasibility. As noted previously, parts are chosen according to the heuristic rule provided by the second part of the particles' position vector, which is assumed to be $X'=(7, 8, 1, 1, 3, 7, 6, 10, 1, 2, 5, 4, 1, 8, 9, 6, 7, 6, 3)$.

The possible candidates for transfer from tour 4 to tour 3 are parts 1, 3, and 4 (parts 2 and 5 cannot be transferred to tour 3, since otherwise workstation capacity would be exceeded). To choose which part to transfer, we use the heuristic associated with the first element of the second part of the position vector. In this case, its value is 7 and thus the part associated with “maximum weight” is chosen (part 4). One bin of part 4 is transferred from tour 4 to tour 3. Tour 4 is still unfeasible, since the weight of all parts is above 19kg. However, no more bins can be transferred to tour 3 as the tour capacity in terms of weight has been reached. Therefore, the next bin(s) should be transferred to tour 2. Due to workstation capacity and weight restrictions only part 1 can be transferred to tour 2. Since all tours are now feasible, the procedure terminates. The initial and final solutions found in this phase (first and second step) are shown in Table A3.

[Insert Table A2.3 Here]

According to Tables A2 and A3, the average inventory in the first and second phases of the algorithm are $AIL_1=0.66$ and $AIL_2=0.58$, respectively. These calculated AIL values demonstrate the effectiveness of the second phase of the algorithm in improving the solution regarding the average inventory value.