Branch and Bound Procedures for solving the Assembly Line Worker Assignment and Balancing Problem. Application to Sheltered Work Centres for Disabled.

Cristóbal Miralles*, José Pedro García, Carlos Andrés, Manuel Cardós

Departamento de Organización de Empresas
Universidad Politécnica de Valencia
Camí de Vera s/n, 46071 Valencia, Spain

Abstract

In this paper a new problem called Assembly Line Worker Assignment and Balancing Problem (ALWABP) is introduced. This problem arises in those assembly lines where we have certain limited resources available (normally workers) in which the operation time for every task is different depending on who executes the task, and where there are also some task-worker incompatibilities defined. The problem consists of providing a simultaneous solution to a double assignment: (1) tasks to stations; and (2) available workers to stations.
After defining the mathematical model for this problem, a basic Branch and Bound approach with three possible search strategies and different parameters is presented. We also propose the use of a Branch and Bound-based heuristic for large problems and analyze the behaviour of both exact and heuristic methods through experimental studies. Finally the implementation of these procedures in a Sheltered Work centre for Disabled -the real environment which has inspired this research- is described. In these centres the adoption of assembly lines provide many advantages, since the traditional division of work in single tasks may become a perfect tool for making certain worker disabilities invisible. Efficiently applying this configuration helps these centres to achieve their primary aim: growth in order to provide more jobs for more disabled people, but always considering the specific limitations that the disabled workers have. In this sense this paper shows one of the possible real applications where Operations Research can help not only to get economic and productive benefits but also certain social aims.

Keywords: Assembly Line Balancing, Mathematical Programming, variable processing times, disability

1. Introduction

There are about 386 million disabled people between the age of 16 and 64, normally with very high unemployment rates from 13% to even 80% in certain countries. Current practices for the treatment of the physically and/or mentally handicapped prescribe meaningful job activity as a mean towards a more fulfilling life and societal integration [4]. In many countries, these practices have facilitated the development of many *Sheltered Work centres for Disabled* (from now on SWD) where disabled people can get a job in the same way as any other person. This model tries to get away from the traditional stereotype that considers disabled people as not able to develop a continuous professional work. Just as any other firm, a SWD compete in real markets and must be flexible and efficient enough to adapt to market variations. The only difference is that the SWD is a Not-For-Profit organisation. Thus, the potential benefit that may be obtained from being efficient usually improves the growth of the SWD. This means: more jobs for disabled people, which is in fact the real primary aim of every SWD.

In these centres the adoption of assembly lines provide many advantages, since the traditional division of work in single tasks may become a perfect tool for making certain worker disabilities invisible. In fact, an appropriate task assignment can even become a good therapeutic method for certain disabilities

* Corresponding author. E-mail address: cmiralles@omp.upv.es

rehabilitation. But some specific constraints relative to time variability arise in this environment, and then the balancing procedures applied in this environment should be able to reconcile the following objectives (that should no longer be seen as contradictory but complementary): (1) to maximise the efficiency of the line by balancing the workload assigned to each available worker in every station; (2) to satisfy and respect the existent constraints in this environment due to the human factors when assigning tasks to workers. After analysing many SWD involved in our R&D project, these specific constraints have been summarized as follows:

- There is usually a great difference among the deterministic mean operation times for each task depending on which worker executes it.
- In many cases, the task time is not only high, but also this specific task is directly impossible to be developed for some disabled workers (this is quite usual when talking about physical or sensorial disabilities).
- Apart from this incapability of some workers to carry out certain tasks, there are also some task-worker assignments that should be considered *a priori* by therapeutical or other specific reasons (e.g. for certain mental disabilities the work routine is sometimes recommended by psychologists as a therapy for the establishment of certain behaviour habits [14]).
- The own variability of operation times for a disabled worker is higher than usual. Normally there are not generically slow or speedy workers. Instead, workers can be very slow, or even incapable, when executing certain tasks, but very efficient when developing some others. Furthermore, environmental factors affect to their fitness and yield, so that a control of such variations is desirable to maintain reliable input data.
- Some specific disabilities need some special treatment also when assigning these workers not only tasks but stations. For example, when there is some orally or audibly inhibited individual is better to assign him or her to the first or last station, because normally working in the centred stations needs a higher level of coordination and communication (although sometimes the opposite can also be desirable in order to improve their abilities).
- The primary SWD aim is to promote a work environment that helps disabled workers to have a positive and constant evolution in their own capabilities, in order to integrate them as soon as possible in ordinary Work Centres. So, it is usual that many workers leave the SWD when they reach their best yields. The SWD must then replace them with new workers; which depending on their disabilities, will make redesign of the existent work assignments necessary.
- Absenteeism is also very common in this environment, since disabled workers have more health problems than usual.
- Also, periodic psychological support and control is mandatory in SWD. This, and the last two circumstances, justify even more the design of agile resolution procedures for SWD assembly lines.

This paper shows the results of the work done for modelling and solving these circumstances that are not often considered in the Assembly Line literature. Focusing on our problem, this basically happens in real situations where we have certain resources available, and where processing time is very different depending on who executes the task; so that this time can not be considered fixed. This is usual in SWD but also in some robotic lines, where a task can be assigned to different available machines with different processing times. Therefore, for every task it is necessary to define different processing times for every worker or server and the problem consists of providing a simultaneous solution to a double assignment: (1) tasks to stations, as in classical *Simple Assembly Line Balancing Problem* (SALBP); and (2) available workers to stations.

1.1. Paper structure

The paper is organized as follows: first we highlight our assumptions for the problem introduced. A review of the related assembly line balancing approaches in the literature is then presented. After formulating a mathematical model for this new problem, a basic Branch and Bound approach with three possible search strategies and different parameters is presented. This section is completed with a comparison of all the variants of the procedure, testing them against a set of self generated problems. This is achieved through a two-level three-factor full factorial experimental study, whose main conclusions are reported. In next section we propose the use of a Branch and Bound-based heuristic for large problems and also analyze its behaviour through a similar experimental study. Finally, the implementation of these procedures in a real SWD is described and some conclusions and further research are exposed.

2. State of the art

Assembly line balancing problems have been usually faced from an academic point of view without considering several realistic requirements, constraints and specific conditions that are often present in many industrial environments. Therefore, a literature review has been provided in order to find those assembly line balancing approaches most related to the problem introduced. But before reviewing these references, certain basic assumptions must be stated in order to completely define our problem:

1. Tasks processing times and precedence relationships are known deterministically.
2. A single product is assembled on the line.
3. We define a serial paced line where buffers are not considered.
4. There are certain workers available, where task processing time can be different depending on which one of the workers executes the task (since the workers have different abilities and capabilities).
5. There are not generically slow or speedy workers. Instead, workers can be very slow, or even incapable when executing some tasks, but very efficient when executing some others.
6. Every worker is assigned to only one workstation.
7. Every task is assigned to only one workstation, provided that the worker selected for that station is capable of performing the task, and that the precedence relations are satisfied.

2.1 Literature review

In its basic form, an Assembly Line consists of a finite set of work elements or single tasks, each having an operation processing time and a set of precedence relations, which specify the permissible orderings of the tasks. The fundamental assembly line balancing problem is to assign the tasks to an ordered sequence of stations, such that the precedence relations are satisfied and some measure of effectiveness is optimized [9]. Assembly Line literature is mainly based on fixed operation times. This simplification is only justified in those cases where operation time variation is small enough. However, few references consider deterministic task times that are different depending on which worker executes the task, despite the fact that this is a quite common situation in many real assembly lines.

[15] proposes a heuristic for the ALBP resolution considering variable operator performance levels. [1] consider how workers must be located when they operate at different speeds, but inside a specific pseudo line configuration called TSS (Toyota Swen System), in which the workers are the ones that carry the product along the different stations where successive operations are executed. [7] study the design of an unpaced line, considering different workers abilities and that overtime can be used when the daily production quota is not reached. [13] study a case in which there are speedy and slow workers (independently of the assigned task) but their solution starts from an already balanced line and their only objective is to temporarily rotate the workers. [8] study a case where workers are generically fast or slow for every task assigned to them. But also they start from an already balanced line where the goal is to efficiently rotate the workers and not to balance the line. More recently [6] present a model for the ALBP that considers variable duration of task times depending on the station assigned, applicable to several industrial situations found where it is necessary to differentiate worker types, but only between two kinds: experts and inexpert individuals.

Other studies address the problem in which there are more than one type of machine. When the decision problem of selecting processing or equipment alternatives is combined with the balancing problem, models incorporate costs and/or profits that must be optimised. In these cases the name *Assembly Line Design Problem* (ALDP) is frequently used in the literature [3]. [10,18] refer to the problem as *Assembly System Design Problem* (ASDP). In this sense [10] consider a model where the stations to be installed at an assembly line are chosen from a set of nonidentical station types with different equipment. However, the balancing problem is simplified by assuming a fixed task sequence (serial precedence graph). [19] consider a process which may be complemented by one or more optional process alternatives. These optional alternatives create fixed costs per time unit. Due to a desired production rate, a lower bound on the cycle time is given, which is reached through a branch and bound procedure that selects/discards single alternatives and computes upper and lower bounds by solving respective SALBP instances. [5] consider equipment alternatives and minimize the total equipment costs for a given cycle time. Every station is provided with one equipment chosen from a set of equipment types. Each type has individual costs and an individual influence on the task times. So two problems arise: (1) A variable number of stations need to be installed and provided with equipment. (2) The tasks have to be assigned to the

stations considering some assignment constraints. The branch and bound procedure designed is based on task-oriented construction scheme and uses a Minimum Lower Bound strategy. The procedure is capable of solving problems of moderate size, therefore a heuristic version of the procedure is developed which skips nodes of the branch and bound tree controlled by an input parameter. The same problem is examined by [16] who propose a Dynamic Programming procedure and a branch and bound procedure. [21] consider a restricted version of the problem related to robotic assembly lines, where all equipment types have identical costs. [17,18] propose branch and cut procedures for basic and generalized ASDP. Also an ASDP where an operating mode defining the task times and equipment costs has to be chosen for each task is considered by [20].

Although some of these references also face a similar double assignment of tasks and resources to stations, many differences arise. The problem presented here is not a cost problem in which there are alternative machines with different costs and the total cost has to be minimized. In all cases the fundamental difference is that in our problem, the resources available are constrained. There are unique workers that can only be assigned once. In some cases there exist workers with similar characteristics, but even in these cases there is not an infinite number of workers available, as assumed in ASDP problems. Therefore this is a different problem that has been named *Assembly Line Worker Assignment and Balancing Problem (ALWABP)*, and that requires new modelling approaches that will be presented.

3. Problem Formulation: The Assembly Line Worker Assignment and Balancing Problem

From our experience, the initial problem we face is usually to maintain an exhaustive control of the operation time for each task for every worker. Without this control, any modelling/resolution proposal is worthless. Then, assuming that such control exists, and according to the assumptions made in section 2.1, the most typical situation in SWD is to have certain workers available (each one of them with operation times defined for every task) and where efficiency on the line must be maximised, which means: cycle time has to be minimised.

Then, IP model for this problem will be defined using the following notation:

i,j	Task
h	Worker
s	Workstation
N	Set of tasks
H	Set of available workers
S	Set of Workstations
A	Set of assignments a priori (i,h) task- worker
I	Set of incompatible assignments task-worker (i,h) due to therapeutical reasons
Z	Set of assignments a priori (h,s) worker-station
C	Cycle Time
m	Number of worksations
p_{hi}	Processing time for task i when worker h executes it
$lowp_i$	Lowest processing time for task i from all available actual workers
Dj	Set of tasks immediately preceding task j in the precedence network
x_{shi}	Binary variable equal to 1 only if task i is assigned to worker h in station s
y_{sh}	Binary variable equal to 1 only when worker h is assigned to station s

Table 1- Notation for ALWABP

According to this notation we have the following formulation:

$$Min \quad z = C \tag{1}$$

subject to:

$$\sum_{h\in H}\sum_{s\in S} x_{shi} = 1; \qquad \forall i \in N \tag{2}$$

$$\sum_{s\in S} y_{sh} \le 1; \qquad \forall h \in H \tag{3}$$

$$\sum_{h \in H} y_{sh} \leq 1; \qquad \forall s \in S \qquad (4)$$

$$\sum_{h \in H} \sum_{s \in S} s \cdot x_{shi} \leq \sum_{h \in H} \sum_{s \in S} s \cdot x_{shj} \qquad \forall i, j \,/\, i \in D_j \qquad (5)$$

$$\sum_{i \in N} p_{hi} \cdot x_{shi} \leq C; \qquad \forall h \in H;\ \forall s \in S \qquad (6)$$

$$\sum_{i \in N} x_{shi} \leq M \cdot y_{sh}; \qquad \forall h \in H;\ \forall s \in S \qquad (7)$$

with:

$$y_{sh} \in [0,1] \forall s \in S, h \in H$$

$$x_{shi} \in [0,1] \forall s \in S, h \in H, i \in N$$

$$M > \sum_{h \in H} \sum_{i \in N} p_{hi}$$

Where:

- Objective function (1) minimizes the cycle time.
- With constraints set (2) every task *i* is assigned to a single station *s* and worker *h*.
- (3) and (4) ensure that every worker can be assigned to an only one station, and that in every station there is only one worker.
- Constraints set (5) reflects the precedence relationships between tasks *i* and *j,* where *i* is predecessor of *j*.
- (6) and (7) imply that every worker *h* assigned to station *s* can have more than one task, whenever given cycle time *C* is not overcome. As Cycle Time *C* and y_{sh} are both variables, (6) and (7) are defined separately in order to maintain the model linearity.

In analogy with SALBP-2, where the aim is to minimize the cycle time given a set of workstations, this problem will be named as ALWABP-2, modelling the most typical situation in SWD: given certain unique workers, to minimize the cycle time.

In SALBP-1 the aim is to minimize the number of stations given a target cycle time, and then an ALWABP-1 problem can be also formulated. As this situation is not so usual in this environment, for the sake of brevity, we prefer to focus this paper just in the ALWABP-2 problem, although the procedures presented in Section 4 have a modular design that enables the resolution of both kinds of problem.

3.1 Additional constraints more specific of SWD

The model exposed is suitable for many real assembly lines where diversity in processing times exists. Therefore this model can help managers to achieve a better assignment; focusing not only on balancing tasks but also on assigning workers to stations properly. In the specific case of SWD we can find some more specific features that should be added to the model as new constraints whenever they exist:

- Apart from the incapability of some workers to carry out certain tasks, there might be also some task-worker assignments that must be considered *a priori* by therapeutic or other specific reasons. This would add the following constraint:

$$\sum_{s \in S} x_{shi} = 1; \qquad \forall (i,h) \in A \qquad (8)$$

- As it has been said, some specific disabilities need some special treatment also when assigning these workers to stations. For example when there is some orally or audibly inhibited individual sometimes is better to assign him or her to the first or last station,

because normally working in the centred stations needs a higher level of coordination and communication. In those cases:

$$y_{sh} = 1; \qquad \forall (s,h) \in Z \qquad (9)$$

- The SWD aim is not only productive benefit but also social benefit of integrating disabled people. Due to this philosophy, sometimes is desirable that all workers must have at least one task assigned, even the slowest ones. Therefore, in these situations the constraint (10) may be added to the ALWABP-2 model in order to avoid any worker being not assigned:

$$\sum_{i \in N} \sum_{s \in S} x_{shi} \geq 1; \qquad \forall h \in H \qquad (10)$$

4. A branch and bound approach for solving ALWABP

In the last decades, many branch and bound approaches have been proposed in the literature for solving different combinatorial problems, including assembly line balancing [22]. In this research, starting from a basic task-oriented branch and bound approach, different variants have been developed for solving the ALWABP-2 problem. Every one of these procedures has a different behavior depending on three basic parameters:

- search direction (*forward*, starting from initial tasks with no predecessors; or *backward*, starting from the tasks with no successors)
- priority rules applied for selecting nodes from a candidates list
- search strategy applied

Before describing these parameters, two common features for all the procedures will be described in the next section: first an overview of the general philosophy of the Branch and Bound procedures will be done; this will be followed by a description of the node branching process and the lower bound calculated for every node.

4.1 *General description*

Most of SALBP-2 resolution procedures in the literature are search methods based on repeatedly solving instances by SALBP-1 procedures [23]. The same way, the procedure developed for solving ALWABP-2 problem is somehow based on an ALWABP-1 approach. This procedure always explores the solution space trying to find the assignment with less number of stations. The first attempt is done with a starting cycle time C, and while C is unfeasible (where C is too low because more than the available workers are needed) it is iteratively increased by one. The first time that C is possible, the assignment will normally include all workers available, and this will be the optimal solution for the ALWABP-2 with a minimum cycle time. In this sense, the starting cycle time for all the procedures is set to C = max (C1, C2), since cycle time should accomplish the following statements:

- A cycle time may never be less than the minimum task time, since tasks are indivisible. In this case, in the best assignment that could be achieved, every task would be assigned to the worker that performs it fastest. Then, being $lowp_i$ the lowest processing time for every task i, we have:

$$C \geq C1 = \max(lowp_i) \qquad \forall \mathrm{i} \in \mathrm{N} \qquad (11)$$

- On the other hand, if we relax the problem by ignoring the precedence constraints and by expecting a perfect assignment -in which every task is assigned to the worker with the lowest processing time- we can adapt the bound defined in [2] for SALBP, which is obtained by using an analogy with the Bin Packing Problem. In our case, this bound is:

$$C \geq C2 = \left\lceil \frac{\sum_{i \in N} lowp_i}{H} \right\rceil \quad (12)$$

4.2 *Branching process and Lower Bound in every node*

In all the procedures designed both the branching of nodes and the calculation of Lower Bound for every branched node, follow the same outline. Regarding the branching there are two different situations to manage throughout the process:

- Starting in node 0, when a new station is opened, one node for every feasible combination of workers and task available is created.
- Once inside a station, only feasible tasks for the worker actually assigned to that station are selected and one node is created for each one of these tasks.

The Lower Bound (LB) is calculated for every branched node in order to give priority to nodes that are potentially better according to the search strategy applied. The Lower Bound for every node will be the minimum number of stations that could be achieved following that node. In our case it will be composed of two elements:

$$LB = K + \left\lceil \frac{\sum_{i \in G} p_{ig} + \sum_{i \in \sigma} lowp_i}{C} \right\rceil \quad (13)$$

With:

K = actual number of stations
g = worker selected in this node.
G = set of tasks already assigned to actual worker g in actual open station, including the task just selected in this node.
σ = set of tasks still not assigned.
$lowp_i$ = lowest processing time for task i from all still not assigned workers.

To compute the Lower Bound in every open node, first the actual number of stations in taken with K, which is an element associated to past decisions only. Then, with the second term, we relax the rest of the problem by (1) ignoring the precedence constraints; (2) expecting the best situation we could have; which would be that for every future task-worker assignment the processing time would be $lowp_i$. Therefore it is not necessary to mind about the station being just closed (all feasible worker and task combinations are branched) or not (only feasible tasks for the worker selected are branched). The calculation of the Lower Bound is the same in all cases.

Once introduced the common features for all the procedures, the three different search strategies designed for opening and revisiting nodes throughout the process will be described next sections. The three strategies defined have been:

- Depth First Search with Complete Node Development
- Best First Search
- Minimal Lower Bound

4.3 *DFSC: Depth First Search with Complete node development*

The DFSC strategy will be described through the diagram in Figure 1. As we can see in this figure, starting from node 0, for every node we will have two different situations when branching: (BN) if we are opening new station; or (BA) if we are in an open station.

Once branched the nodes, two filters are applied to all unexplored nodes:

- F1-minLB: only nodes whose lower bound is minimal (LB = minLB) pass this filter.
- F2-DEEP: only nodes that belong to the deepest level in the partial solution tree generated pass this filter.

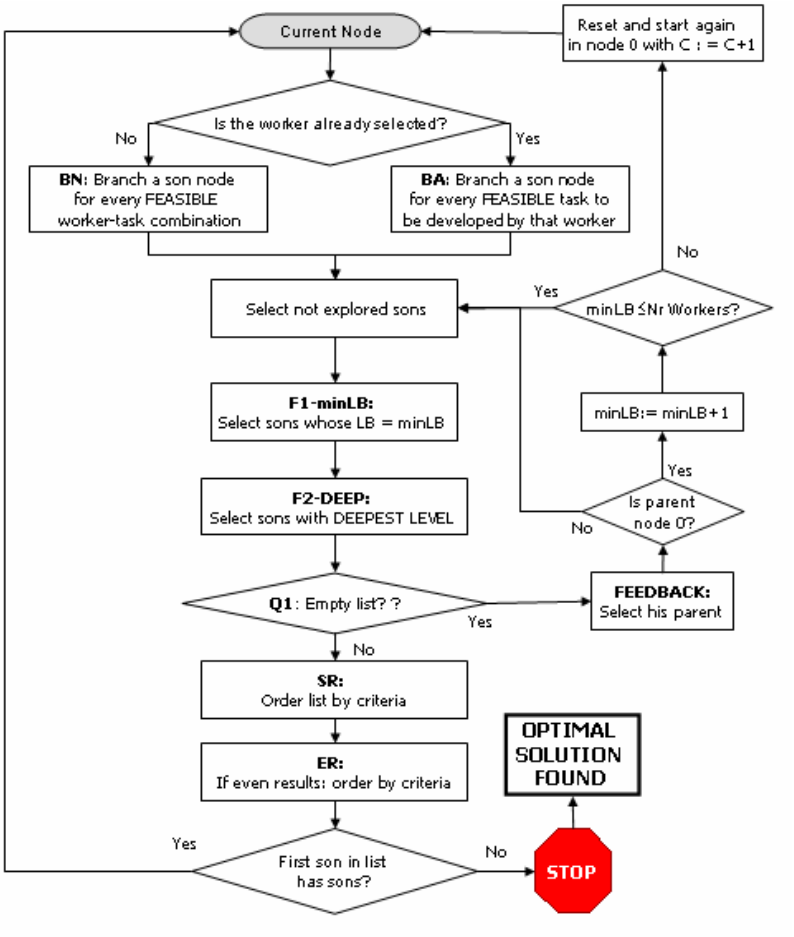

Figure 1.Diagram for the DFSC procedure

All the filtered nodes are then sorted by certain Selection Rule (SR), and first one is selected as the current node, and then the process starts again. Whenever it is necessary, an Even results Rule (ER) is also applied. The three criteria defined for being used as selection/even results rule are:

- First the node with lower process time of the task just selected.
- First the node with higher number of successors of the task just selected.
- First the node the task belongs to the mean faster worker (useful only when opening new station, where we have nodes that belong to different workers).

This branching, filtering and sorting process is iteratively done while there are nodes to sort in Q1 (see Figure 1). But when there is an empty list in Q1, the current node parent is revisited and then alternative branches are explored, filtered and sorted. If in this feedback process node 0 is reached, this means that all the alternative ways have been already explored, and it is necessary to increase minLB by one, since the actual one is not possible. Then many nodes that were discarded are now reconsidered.

But when doing this, if we have a situation where the increased minLB becomes higher than the number of workers available, this means that actual cycle time is too low and not feasible. Therefore the cycle time is increased by one and the algorithm starts again. The procedure iterates this way making all necessary returns to ancestor nodes in order to guarantee an optimal solution. The stop condition will be that no son node exists for first selected node sorted by SR and ER (all tasks already assigned). This means that this node is a leaf of the solution tree and the final assignment provided by this node is optimal.

4.4 ***BFS: Best First Search strategy***

As we can see in Figure 2, the behaviour of this strategy is identical in the branching process, but in this case the filters applied to the unexplored nodes are F1-minLB (the same as in DFSC) and F2-LOW, which filters only those nodes that belong to the lowest level in the tree. This way the solution is being

constructed level by level, taking in every step the best node and never going ahead until all the nodes in actual level have been branched. Therefore, there is no need to define any mechanism for revisiting ancestor nodes: when we have an empty list in Q1, minLB is directly unfeasible and is increased by one.

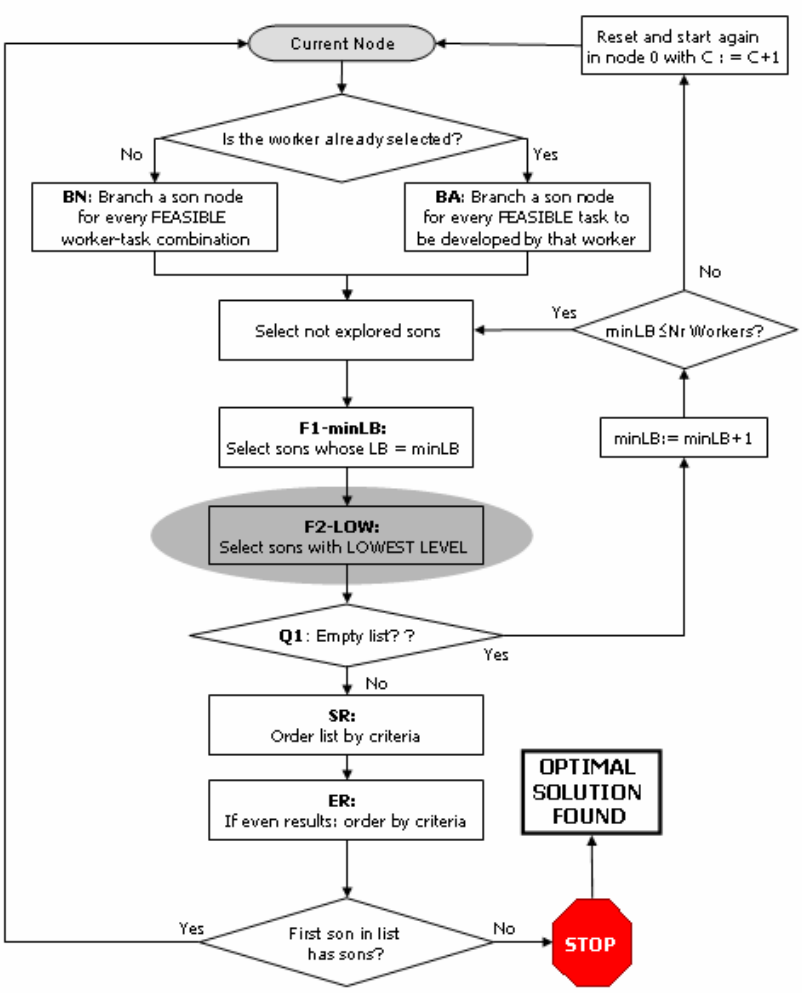

Figure 2.Diagram for the BFS procedure

4.5 ***MLB: Minimal Lower Bound strategy***

For the sake of brevity we will describe the third strategy implemented with the same diagram as *Best First Search,* since both strategies are very similar. The behaviour of Minimal Lower Bound strategy is identical excepting the shaded circle in the figure 2. In this case the only filter applied to unexplored nodes is F1-minLB. Then, all the nodes that fulfil this condition, regardless of what level they belong to, will be incorporated to the Q1 list and will be sorted by selection and even results rules.

5. Experimental study for Branch and Bound procedures developed

Since ALWABP is a new problem, there is no standard set of benchmark problems available for testing. So we have constructed a two-level three factors full factorial experimental study based on the *Jackson* problem, from the classical collection of SALBP problems of [12]. From this standard problem the original precedence network was preserved. The original task time was used for first worker and new workers tasks times were randomly generated from first worker ones. From our experience in SWD involved in our R&D project, the range for randomly generating these times should not be greater than three times the original task time. When a worker is over this range for a certain task, we will assume that this task shouldn't be assigned to him/her. The worker will then be assigned infinite time for that task (which means: task not assignable to this worker). In fact, different percentages of incompatibilities in the tasks-workers matrix were also defined for this full factorial study. The problems were generated according to the following three parameters:

- **NrW**: The relation between the number of tasks and the number of workers (size of the matrix).
- **Var**: Variability of task times for the different workers.
- **Inc**: The percentage of task-worker incompatibilities defined a priori.

For the first factor two levels, high (number of tasks 3 times higher than number of workers) and low (number of tasks 6 times higher than number of workers), were defined. The different tasks times for every task *i* were randomly generated from a uniform distribution with range selected according to the original time t_i. The two levels defined for the task times variability used the distributions U[1,t_i] and U[1, $3t_i$] for low and high variability. And finally, the low and high percentage of incompatibilities in the tasks-workers matrix was set to 10% and 20% approximately.

Apart from the levels defined for the problems, the levels defined for the three exposed parameters relative to the procedures are summarized below:

- **Search direction (DIREC)**: Forwards (F) or Backwards (B).
- **Selection rule and even results rule (SR_ER)**: The three criteria exposed in section 4.3
- **Search strategy (STR)**: DFSC (D), BFS(B) or MLB (M)

40 problems were generated and they were run for the 36 different procedures, this leds to 1440 experiments. In order to evaluate which the most efficient procedure is in terms of number of nodes generated for providing an optimal solution, the indicator used was the *Node Perceptual Increment* (NPI) for every experiment defined as:

$$NPI_P = \frac{NO_P - NO_{BestP}}{NO_{BestP}} \cdot 100 \qquad (13)$$

Where:
NO_P = Number of nodes generated by the procedure P for solving a problem
NO_{BestP} = Number of nodes required by the best procedure (the one that solved it with less nodes)

5.1 Results

From our ANOVA analysis we may summarize the main conclusions obtained by means of Fisher Test Graphics. The main statistically significant factors were the strategy and the direction, being indifferent the criteria applied both for selection and even results rules:

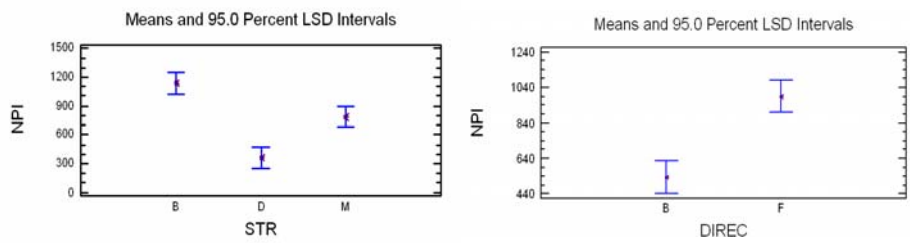

Figure 3.Fisher Test Graphics for STR and DIREC

As we can see in Figure 3, the DFSC strategy is the one that needed less node development for optimally solving the problems; being the backward direction the one with better behaviour. As the procedure is executed against different kind of problems, according to the full factorial development exposed, also some valid conclusions about robustness can be obtained when considering double interactions between the factors of the problems and the factors of the procedure. In this sense from the ANOVA we got only two main significant interactions:

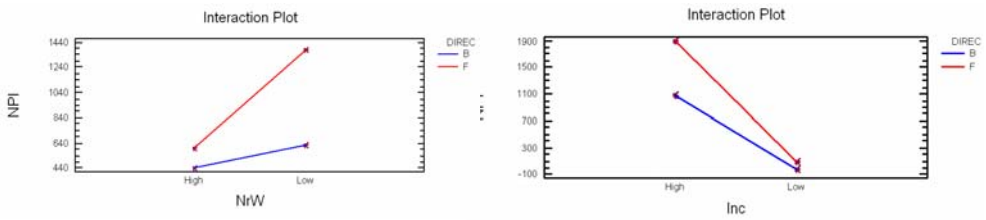

Figure 4. Main significant double interactions

As it is shown in Figure 4, Backwards Direction was generally convenient, however when facing problems with low number of incompatibilities or with high number of workers, to execute backwards or forwards became nearly indifferent (similar results). About the strategy there is no significance: when facing different kind of problems always DFSC is better.

6. Heuristic approach: a multipass algorithm from DFSC

Facing real problems makes the design of heuristics rules desirable in order to get results in reasonable computational time. SALBP is known to be NP-hard [11], and SALBP is a special case of ALWABP where every task has a fixed duration. Therefore ALWABP is also NP-hard and it is fully justified to develop heuristic solving methods in order to achieve good results in a reasonable computational time, especially in SWD where different circumstances appear: in these centres absenteeism is high as disabled workers have more health problems than usual. Additionally psychological support and control is also mandatory in SWD. Therefore, just at the beginning of every working day the manager knows exactly what workers are available, and fast effective algorithms that provide good solutions are desirable for an early daily assembly line set up.

Although some other research lines are now being developed, for designing heuristic procedures the first option considered was to modify the Branch and Bound procedures in order to save some nodes generation even if we do not ensure an optimal solution. In the case of DFSC procedure this has been done just by neglecting some back step condition, and also by trying to diversify the solution space explored (for avoiding local optimums). This multipass heuristic designed will be described using a similar diagram to DFSC where modifications are highlighted:

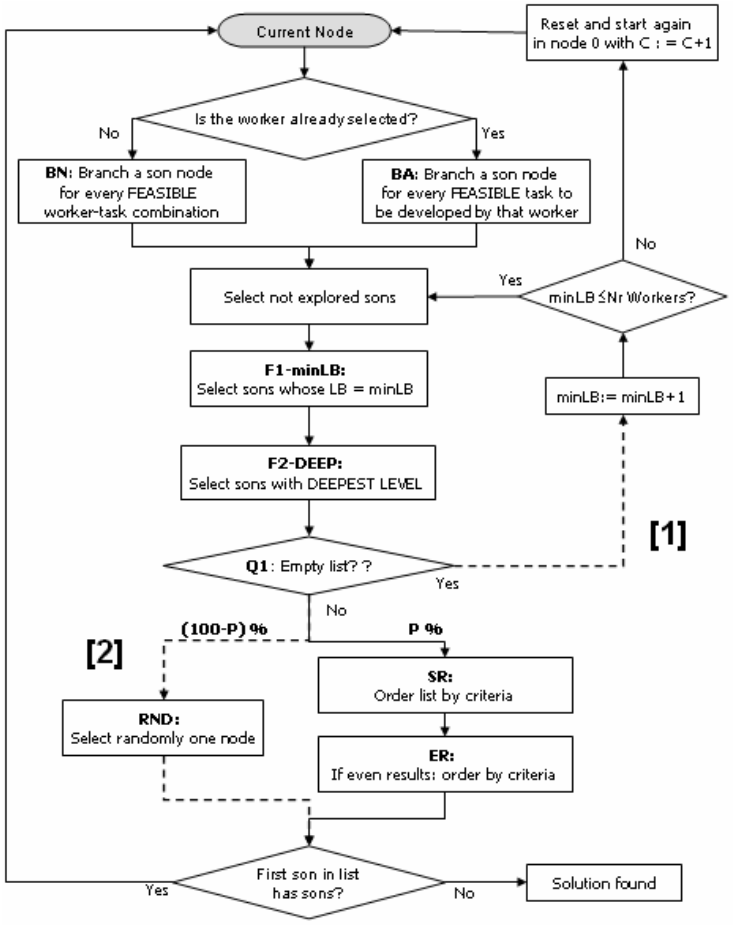

Figure 5.Diagram for the heuristic procedure designed

On one hand we notice how we avoid revisiting ancestors (see modification [1] in Figure 5): whenever there is an empty list in Q1, minLB is directly increased by one and the algorithm proceeds until a final solution is obtained.

On the other hand (see modification [2] in Figure 5) the procedure is a multipass algorithm that makes different runs during a certain predefined time t. Then, in order to diversify the solution space visited in the different runs during t, a percentage of times (100 –P) % the node is selected randomly from the list, while P% of times the node selected for branching is the first on the list, as with the DFSC algorithm.

6.1 Experimental study for the heuristic

For this study 40 new problems have been generated starting from the classical *Mitchell* problem [12] using the same two-level three-factor full factorial experimental design approach described in section 5. Every problem has been solved with different values for the parameters *P* (diversification) and *t* (runtime). P has been set to two fixed values of 10% and 40%, while the runtime t has been defined according to the size of the problem in order to obtain general conclusions. In this sense, time of execution is set in every experiment according to the number of tasks (N), and the number of workers available (H) in the problem. Starting from a time $t = 0.25 \cdot H \cdot N$ four different levels of time are defined, where every new level is twice the amount of time at the previous level. Then we have eight different combined levels for these parameters:

- H-DFS with 1t and P=10% (1t_P10)
- H-DFS with 1t and P=40% (1t_P40)
- H-DFS with 2t and P=10% (2t_P10)
- H-DFS with 2t and P=40% (2t_P40)
- H-DFS with 3t and P=10% (3t_P10)
- H-DFS with 3t and P=40% (3t_P40)
- H-DFS with 4t and P=10% (4t_P10)
- H-DFS with 4t and P=40% (4t_P40)

Before launching the experiments, the optimal solution was obtained for every problem, so that it can be compared to the solution provided for each heuristic. In this sense the indicator used for this comparison in every experiment was the *Solution Quality Perceptual Decrease* (SQPD), defined as:

$$SQPD_{Heu} = \frac{Sol_{Heu} - OptSol}{OptSol} \cdot 100 \qquad (14)$$

Where:

OptSol = Optimal cycle time
Sol_{Heu} = Cycle time provided by the heuristic

6.2 Results

From the ANOVA (analysis of variance) done with the results of the experimental study we obtain the following graphic that shows how the results converge quite quickly to SQPD's of around 10%, which is a reasonable value:

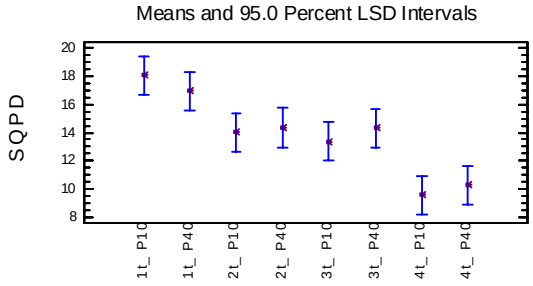

Figure 6. Fisher Test Graphic for the different combined levels defined

This and the fact that the algorithm runtime can be set a priori are specially appreciated in SWD. As it has been explained, due to the high absenteeism existent in this environment, information about the available workers is known just at the beginning of every working day. Therefore effective algorithms whose

runtime is known in advance are really desirable, so that the daily assembly line assignment can be set up on time.

About the double interactions we have noticed a quite robust behaviour, where better solutions are obtained when problems have low variability of tasks times (Var) and low percentage of incompatibilities (Inc):

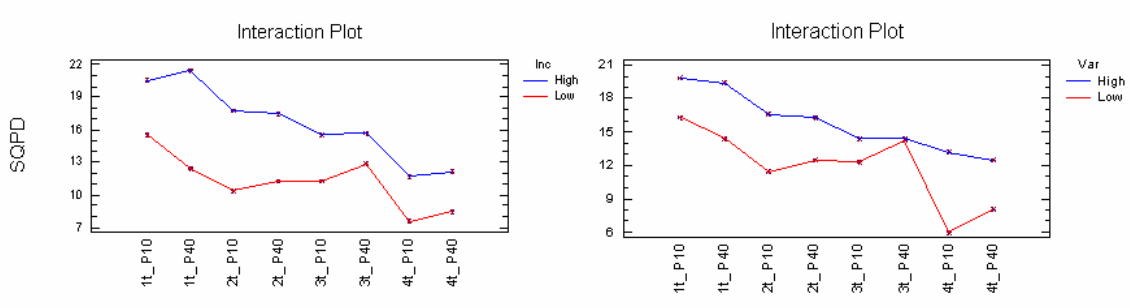

Figure 7. Main significant double interactions

7. Application to real case

Our proposal has been applied in a SWD involved in our R&D project whose main industrial activities are related to assembling of electronic components. In the section of this SWD where we carried out our research, the work places used to be individual before adopting the definitive assembly line configuration. For this reengineering process an effort was made to define properly every task time for every single worker. At first, traditional Work Measurement methods were supposed to be useful for defining these times, but some problems arose and these methods needed some adaptation, whose description is outside of the scope of this paper.

A first prototype assembly line for a product with high demand (involving 7 workers) was designed and successfully implemented; noticing many advantages compared with the former individual workplaces. The definitive data file for this real ALWABP-2 problem was composed of 18 tasks, with about 20% of incompatibilities and there were 22 precedence relations defined and some other constraints such as those commented in section 3. In this new prototype assembly line our balancing methods were used as a Decision Support System for the manager. Finally, adopting the worker assignment and balancing solutions provided showed how assembly lines can be a powerful tool in this special environment, reaching many objectives simultaneously: (1) raising production efficiency; (2) and also making certain disabilities become invisible, so that more disabled people may be integrated into a work team like anyone else. This is only possible through resolution methods like those proposed here, which consider both aims simultaneously.

In fact, after this successful experience, the new challenge for the SWD has been the recent launching of four new assembly lines, where some workers from other sections, that so far were incapable for the former individual workplaces, are now assembling these electronic products. Furthermore the staff has been significantly increased and the primary objective of the SWD has been then fulfilled: now more than twenty disabled people have a new job. In this new more dynamic scenario, our heuristic methods are even more necessary, and are being successfully applied. In any case much research is still necessary since new requirements arise.

8. Summary and further research.

Fortunately a great number of Sheltered Work Centres for disabled people have been created in Spain in the last two decades. Although they get some institutional help, these centres have to survive in real markets and then need to be efficient. Only being efficient they can reach their primary aim: grow in order to provide more jobs for more disabled people. In this sense the assembly lines are very useful in this environment since their traditional division of work in single tasks can make some disabilities disappear, just by finding a proper assignment of tasks to workers and workers to stations.

The special features typical for this environment have been analysed, and a mathematical model for the *Assembly Line Worker Assignment and Balancing Problem* that arises has been presented. This model

takes into account the specific characteristics of these centres. A basic Branch and Bound approach, with three possible search strategies and different parameters, has been presented and tested through an experimental study that has shown an overall better behaviour of the *Depth First Search with Complete node development* strategy. Simply by properly modifying this procedure, a multipass heuristic has been developed. Through another experimental study, the quality of the solutions provided by this heuristic has been reported showing reasonable values. Finally, a brief description of the application of these procedures to a real case has been presented, evidencing its potential benefits in the reality.

Further research includes three main topics: in a first stage the design of efficient job rotation procedures. Job rotation is always desirable but here is even more important, since it can contribute to the improvement, if not simply fundamental maintenance, of certain worker abilities. As the task times are different depending on the worker, these procedures are not as obvious as in ordinary assembly lines. Another interesting research line, which would widen the scope of the problem, is considering parallel workstations, which would enable more combinations of assignments. Both modelling and design of solving procedures should be faced in this new scenario. Finally adapting some metaheuristics, like Genetic Algorithms or Tabu Search, for being applied to the problem seems to be quite interesting due to its high combinatorial nature.

Acknowledgments

This work received the support of the Polytechnic University of Valencia through the Research Incentive Program PPI-00-04. We like to acknowledge the very helpful comments of Professor Steve Disney of the Cardiff Business School. Finally we would like to thank the SWD involved in this research for their collaboration. Thanks to: SERVIDISCA, ADAPTA, ESPURNA, CEEME, EUROCEBOLLA and ALCER-TURIA.

References

[1] Bartholdi, J.J. and D. Eisensteein, 1996. A Production Line that Balances Itself, Operations Research 44, 21-33.

[2] Baybars, I., 1986. A survey of exact algorithms for the simple assembly line balancing problem, Management Science 32, 909-932.

[3] Becker, C. and A. Scholl, 2003. A survey on problems and methods in generalized assembly line balancing, Jenaer Schriften zur Wirtschaftswissenschaft 21/03, FSU Jena

[4] Bellamy, G.T., R.H. Horner and D.P. Inman, 1979. Vocational Habilitation of Severely Retarded Adults: A direct Service Technology, University Press, Baltimore.

[5] Bukchin, J. and M. Tzur, 2000. Design of flexible assembly line to minimize equipment cost. IIE Transactions 32, 585-598.

[6] Corominas, A.; R. Pastor. and J. Plans, 2003. Línea de montaje con tiempos dependientes del tipo de operario 27 Actas Congreso Nacional de Estadística e Investigación Operativa. 2789-2795

[7] Doerr, K.; T.D. Klastorin and M.J. Magazine, 2000. Synchronous Unpaced Flow Lines with Worker Differences and Overtime Cost, Management Science 46, 421-435.

[8] Gel E.S., W.J. Hopp and M.P.Van Oyen, 2002. Factors affecting opportunity of worksharing as a dynamic line balancing mechanism, IIE Transactions 34, 847-863

[9] Ghosh, S. and R.J. Gagnon, 1989. A comprehensive Literature Review and Analysis of the Design, Balancing and Scheduling of Assembly Systems, International Journal of Production Research 27, 637-670

[10] Graves, S.C. and B.W. Lamar, 1983. An integer programming procedure for assembly system design problems, Operations Research 31, 522 - 545.

[11] Gutjahr, A. and G.L. Nemhauser, 1964. An algorithm for the line balancing problem. Management Science 11, 2

[12] Hoffmann, T.R., 1990. Assembly line balancing: A set of challenging problems, International Journal of Production Research 28, 1807-1815.

[13] Hopp, W.J., E. Tekin and M.P. Van Oyen, 2001. Benefits of skill chaining in Production Lines with Cross Trained workers. WorkSmart Laboratory Technical Report.

[14] Jurado de los Santos, P., 1999. La transición escuela/trabajo y la evaluación de personas con discapacidad, Ed. Aljibe, Madrid.

[15] Mansoor, E.M., 1968. Assembly Line Balancing: A heuristic Algorithm for Variable Operator Performance Levels, Journal of Industrial Engineering 19, 618.

[16] Nicosia, G., D. Pacciarelli and A.Pacifici, 2002. Optimally balancing assembly lines with different workstations, Discrete Applied Mathematics 118, 99-113.

[17] Pinnoi, A. and W.E. Wilhelm, 1997. A family of hierarchical models for assembly system design, International Journal of Production Research 35, 253-280.

[18] Pinnoi, A., Wilhelm, W.E., 1998. Assembly system design: A branch and cut approach, Management Science 44,103-118.

[19] Pinto, P.A., D.G. Dannenbring and B.M. Khumawala, 1983. Assembly line balancing with processing alternatives: an application, Management Science 29, 817 - 830.

[20] Rekiek, B., de Lit, P., Delchambre, A. (2002): Hybrid assembly line design and user's preferences, International Journal of Production Research 40, 1095-1111.

[21] Rubinovitz, J. and J. Bukchin, 1993. RALB – A heuristic algorithm for design and balancing of robotic assembly lines, Annals of the CIRP 42, 497-500.

[22] Scholl, A., 1999. Balancing and sequencing assembly lines, 2nd edition, Physica, Heidelberg.

[23] Scholl, A. and C. Becker, 2003. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing, Jenaer Schriften zur Wirtschaftswissenschaft 20/03, FSU Jena.