CHAPTER 7

THE RESOURCE PLANNING FOR ASSEMBLY LINE[1]

I'd like to develop a system for the optimisation of assembly lines in a mixed model line. The main goal is to support the planner. The planners knowledge should be introduced in the system by an interactive process. The uncertain variables like probabilities, precedence graph are taken for an evaluation and optimisation of the system. After this, the planner has to check the result and give restrictions, which at this time, are new for the system. This interactive process leads to certain knowledge of the system and (hope so) to the optimal solution for the assembly line. If anyone knows an application for the whole problem (or parts of it), please send a mail.

Knut Alicke
Universitaet Karlsruhe
Institut fur Foerdertechnik

Keywords: assembly line design, branch & cut, equal piles, grouping genetic algorithm, hybrid assembly line, logical layout, multiple objective, PROMETHEE II, resource planning.

1. Introduction

The conventional approach to the assembly line design (balancing) problem assumes that the manufacturing methods to be used have been predetermined. However, in practice the assembly line designer has several alternatives manufacturing methods. In general, the choice of a solution is guided by cost, labour, reliability, etc. The principal objective of modern assembly systems is to produce high quality and low cost products. The main goal is, of course, to *maximise the profit* of the company.

---

[1] In the resource planning department of a company, there is a marvellous scene that every planner or any planner's systems developer should consider. As he assigns tasks to stations along an assembly line, we hear a voice repeatedly asking "Is it ok now?" followed each time by the response "I don't know!" As the planner continues going through the different stations, looking for the best assignment dealing with the balancing, cost, reliability, etc. he lost the way. He randomly designs an assembly line, since it was detrimental, he let time to his admirer to repeat the question and he repeated the answer sequence yet again.

As introduced in Chapter 2, the main task of the *resource planner* (RP) is to design a flexible assembly system that will be able to assemble a product at least cost. For manual assembly line the global cost of the line is directly influenced by the number of stations. This why, the main objective of classical *line balancing* is to minimise the number of stations. In general, the assembly line dedicated to *small size* products may be *hybrid*. Indeed, the operations can be executed either manually, by robots or by hard automated equipments (Figure 7.1). These lines are called hybrid assembly line (HAL). In this case, the above reasoning about manual assembly line is not valid anymore. In general, process time and cost depend on the resource used. Given a list of candidate equipments available to complete the operations, the design problem thus becomes to decide which resources to select and which tasks to assign to each resource in order to meet the production requirements at a minimum cost. Thus, the main objective is to minimise the cost of the line by integrating design (cost, stations space, etc.) and tasks issues (cycle time, precedence constraints, availability, etc.).

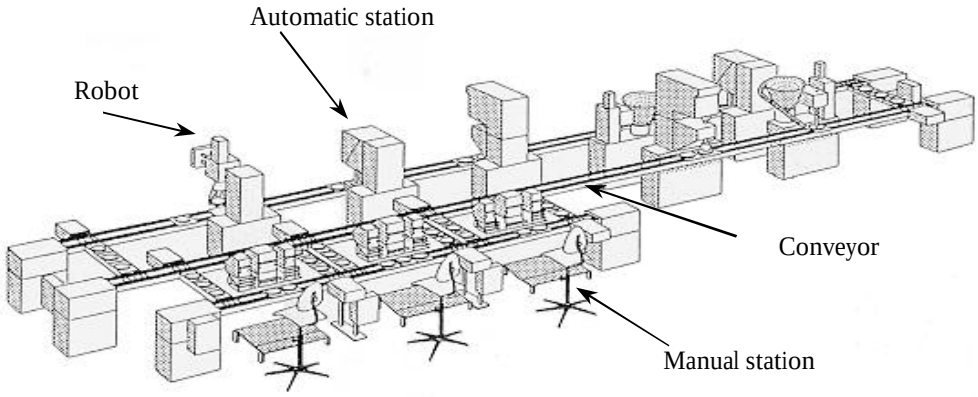

Figure 7.1. Hybrid assembly line.

The term resource planning comes from the fact that the main aim of the method is to assign equipment (*resource*) to task and thus *plan* their utilisation. Many nominations can be found in the literature, the most encountered are: line balancing with processing alternatives, assembly system design or assembly process planning with resource assignment.

The remainder of this chapter is structured as follows. We briefly review work related to ours in section 2. Section 3 is devoted to the explanation of the RP problem we tackle and section 4 to the input data of the problem. A detailed description of the method we propose in done in section 5. A case study will be presented at section 6. We draw conclusions and propose some further works at section 7.

2. State of the art

Most of the research on balancing deal with the so-called SALB problems in which no alternative equipments are considered. That is, each task's process time is fixed,

and the aim is to determine the sets of tasks to be performed on each station. The SALB problem is proven to be an NP-Hard problem, by the way the RP problem is NP-Hard as well. With respect to more than one type of equipment, there are relatively few studies that address this problem.

Graves and Withney (Graves, 1979), in one of the first works on the field, presented a method for the single product equipment selection problem. The approach was based on linear programming techniques and solved by a B&B procedure. The method did not deal with precedence constraints, rather a sequence, and permitted non-serial line layouts in which an assembly unit visits more than once a given station. The goal was to select equipments and make task-assignment so as to minimise the sum of fixed and variable costs.

Graves and Lamar (Graves, 1983), extended the work of Graves and Withney (Graves, 1979) and developed an integer programming cost-based model to the automated system design problem. A column generation procedure was applied to solve the integer formulation of the problem. Their aim was to determine the type and the number of stations, and the operations assigned to these stations. The approach used a fixed assembly sequence[2], which is a first inconvenience, but their main limitation as in (Graves, 1979) was that they permitted non-serial line-layouts. Their model did not explicitly try to balance the total workload across the stations, rather the design criterion is to minimise total system costs, while satisfying a desired production rate. The station loads were often highly unbalanced. As a result of allowing unrestricted floor layouts for the problem, the solutions found were not necessarily physically realisable.

(Gustavson, 1986) developed heuristic methods for solving both the single and multiple product equipment selection problem. The author proposed an alternative to avoid the problem of the non-serial line layouts. These heuristics were also based on a fixed assembly sequence with the inconveniences already mentioned above. Of course, being heuristics these methods cannot guarantee that an optimal solution will be found. One of the reason of Graves and Holmes (Graves, 1988) for developing their optimisation method was the evaluation of the effectiveness of the (Gustavson, 1986) heuristic methods. Tests on single product cases, showed that the two methods found the same solution.

Graves and Holmes (Graves, 1988) considered the design problem with several equipment alternatives, when multi-products are assembled on the same line. They assumed a complete ordering of tasks of the same product and large similarities among different products. These assumptions resulted in a relatively small number of candidate stations and therefore simplified the problem considerably. The method seeks to assign tasks to stations and selects equipments for each of them. It aims to minimise the total cost of the assembly line and it is composed of two steps. The first step consists in enumerating all candidate stations for the system and selecting the

[2] An assembly sequence is an ordering of *m* operations to assemble a product, *m* is the number of components of the product.

least-cost resource type for each candidate station. The second step consists in finding the least-cost assembly system. For this purpose, a graph in which each candidate station corresponds to an arc is used, the length of the arc being proportional to the cost of the station. The least-cost assembly system is then found by solving a shortest-path problem in the network of feasible stations. They addressed this problem, using a method similar to the one of (Gutjahr, 1964). Their algorithm enumerates all feasible stations, selects the best equipment for each, and then chooses the best set of stations. The method guarantees the feasibility of the layout by restricting the system to a linear floor layout. However, given the implicit enumeration of all possible stations, the method is impractical for large problem instances.

Faaland et al. (Faaland, 1992) also used the Gutjahr and Namhauser's heuristic for the ALBP (Gutjahr, 1964). The least-cost assembly system was found by solving a shortest-path problem in the network of feasible stations. The approach is used for building only those nodes of the network which have a reasonable chance to lead to an optimal solution, thus yielding a reasonably fast approximation algorithm. The authors also proposed two other heuristic adaptations of the shortest path procedure that are capable of solving large problems. Their simulation results indicated that the choice of the procedure depends on the problem complexity.

Bard (Bard, 1989) presented a dynamic programming algorithm for solving the ALBP with parallel stations. Solutions represent a trade-off between the minimum number of stations required to achieve a balance and the cost of installing additional facilities. Both equipment cost and task cost are considered. The algorithm takes into account the unproductive time during a cycle.

Pinto et al. (Pinto, 1983) discussed processing alternatives in a manual assembly line as an extension of SALB by relaxing the assumption that all stations are identical. Each processing alternative was related to a given set of tasks. The problem was to decide which alternative (from the existing ones) to use in order to shorten the task duration for a given total cost. Since the line is manual, each task may be performed at any station. The authors proposed an integer programming formulation of the problem. Their solution procedure consisted of a B&B algorithm in which a SALB problem is solved in every node of the tree. Therefore, this method may be used only for a small number of possible processing alternatives. This study presented the interesting advantage of working on a set of precedence constraints instead of a fixed sequence.

Lee and Johnson (Lee, 1991) proposed an iterative method based on integer programming, depth-first B&B and queuing network analysis. The approach allowed to deal with multi-product assembly line. The objective was to minimise the cost of work-in-process inventory, machine investment and maintenance and material handling. They considered five design factors: the number of stations, the number of parallel machines, the tasks assignment, the number of pallets and fixtures and finally the number of automated guided vehicles. The method suffered from a severe restriction: each station consists in a single machine or an identical parallel machines.

Lee and Stecke (Lee, 1996) presented an integrated design support method for flexible assembly systems. The utility of a multi-criteria optimisation tool is underlined in that work. But one of its main weaknesses is that the optimisation was applied only to the cost while satisfying a set of constraints.

In (Pinnoi, 1998) a heuristic method called 'branch and cut' (special kind of branch and bound) was used to solve the problem for single-product assembly line, taking only cost and time parameters into account. It presented the advantage of needing lower computation times compared to the classical B&B methods while producing results within 13% of the optimum on some tests problems.

Tsai and Yao (Tsai, 1993) proposed an integer programming model combined with a simulation adjustment phase. The approach was used to design a flexible robotic assembly line which produces a family of products. Given the work to be done at each station, the demand of each product and a budget constraint, the method determined the robot type and number of robots required in each station along a serial robotic line. Their objective was to minimise the standard deviation of the output rates of all stations, which is the measure for a balanced line.

Rubinovitz and Bukchin (Rubinovitz, 1993) introduced the robotic assembly line balancing (RALB) algorithm for solving single model ALB problems. The method was based on a B&B algorithm and aimed to design (balance) robotic assembly line when several robot types are available. Their model supposed that all the equipment alternatives have an identical purchasing cost. The objective was to find a line balance which minimises the number of stations for a given production rate.

Bukchin and Tzur (Bukchin, 2000) developed an optimal method and a heuristic algorithm to design flexible assembly line when several equipment alternatives are available. Tasks are subject to precedence constraints. The objective was to minimise total equipment cost, given a pre-determined cycle time. They developed an exact B&B algorithm which was capable of solving practical problems. The algorithm's efficiency was enhanced due to the development of good lower bounds, as well as the use of dominance rules to reduce the size of the tree. They also suggested to use a B&B based heuristic procedure for large problems. For large problems that cannot be solved by the optimal algorithm, the authors developed a heuristic procedure. The procedure's control parameters may be chosen according to the size of the problem. The control parameter determines how many nodes of the tree may be skipped, and therefore was responsible for the running time, for the memory requirements, as well as for the distance from optimality of the resulting solution. The method dealt with many realistic features of assembly line, unfortunately it did not deal with the operating mode of tasks which is one of the hardest constraints of the problem.

On the side of interactive and iterative methods, Nevins and Withney (Nevins, 1989) presented a method based on technical and economical considerations. The method helps to construct the cheapest technically feasible assembly line. The method presented the advantage of being complete, but could become very time consuming to apply if the number of operations and the set of usable resources for each

operation become too important. Another weakness was that the assembly sequence was fixed before the application of the method. This could be too restrictive and lead the system to miss the most cost-effective solutions.

Okano (Okano, 1993) proposed a generator of logical expressions able to generate all feasible resource groupings for a given sequence of assembly operations. After obtaining valid combinations of resources, a simulation tool was used to evaluate the different solutions and select the best one. Since the method was based on an assembly sequence, a first fit heuristic method was used to group tasks, where the constraint was the cycle time calculated on the basis of production volume. The criteria for the grouping of operations and resource assignment were the production volume and the production cost. The approach was based on the risky assumption that the constraints of the problem will avoid the combinatorial explosion of the possible solutions set.

Petit (Petit, 1999) proposed a general method for the design of a product and its assembly line. The resource planning is mainly based on two elements. The first element was the 'working principle' (WP), defined as a set of constitutive concepts allowing one to execute a given assembly operation. There could be several WPs by operation. The second element was a matrix of affinities between the different WPs. The author also mentioned a clear need for a object-oriented line design approach based on realistic input data. The main drawback of this approach was the amount of data the designer had to introduce. The method was based on an enumerative algorithm exploring all the combinations of operations and WPs in order to find the best set of stations constituting the optimal assembly line. Solutions are compared on the basis of the affinity inside each station which has to be maximised. The idea of using matrix of affinities including technical criteria was very interesting, even if it seemed difficult to evaluate. The fact of using WPs instead of equipment presented the advantage of reducing the number of alternatives even if the risk of combinatorial explosion of the solution set was present. Another risk linked to the method was that nothing guaranteed that an equipment satisfying simultaneously the technical concept defined by the WP and the operating time needed to comply with the cycle-time constraints did really exist for a reasonable cost. Indeed, one had to remind himself that the indicative times and costs associated to the WPs are just estimations and could be significantly different from the cost and time involved in the real equipment.

Malakooti (Malakooti, 1996) used a multi-criteria decision-aid method for ALB problems where objectives were the number of stations, cycle time, buffer size and the total cost of the operations. For single objective ALBP with buffers, the author used a three steps interactive method to minimise the total cost of the line. The method first balanced the line using the ranked positional weight (RPW) algorithm (see (Helgeson, 1961)). It then looked for the required buffer size using an empirical formulation. Finally the total cost of the line is estimated. For multiple objective ALBP, the author used an additive utility function based on decision maker's preferences (weights). He divided the whole into three simple problems: (1) minimise the number of stations when the cycle time is given, (2) minimise the cycle time

when the number of stations is given and (3) minimise the total cost when the cycle time is given and the number of stations has to be determined. Solving the three problems for different values of the cycle time and the number of stations may lead to different solutions. The best solution from the efficient alternatives was chosen using a 'ranking alternatives by strength of preferences' method. The attributes of the additive utility function were the cycle time, the number of stations, the buffer size and the total cost.

Minzu and Henrioud (Minzu, 1997) presented a general algorithm for the stations determination, for mono-variant products. It was based on the use of a so-called assembly graph, which combines the advantages of the assembly tree and the precedence graph–i.e. it combines the precedence constraints information with geometrical information like the relative orientations of the components. A method of tasks-to-equipment assignment based on this graph was proposed, as well as the operative sequences for the equipment and the computation of the process-time of a station. These methods are very interesting, especially for the evaluation of the process time of a station which allows taking into account the hidden times, the operator moves, and the setup time. The author used the 'constraint propagation' method embedded in the 'Prolog' language and a backtracking approach. A depth-first 'backtracking' method was used during the tasks-to-station phase. However, two drawbacks can be mentioned for these methods. The first one was that they did not take cost and availability factors into account. The second drawback of these methods, already mentioned in (Petit, 1999), was that they had not been linked to an equipment database and were not really considering the technical station design.

McMullen and Frazier (McMullen, 1998) presented a 'simulated annealing' (SA) method to address the ALB with multiple objectives. Two kind of objectives were used: the single and the composite objectives. Three single objectives which are: (1) minimise the design cost associated with both labour requirement and equipment requirement, (2) minimise the smoothness index which intended to distribute work into the station as evenly as possible, and (3) minimise the probability of lateness to deal with the stochastic nature of task durations. They also introduced three composite objectives which are expressed as the weighted sum of the first and the third objective. Different weights were attributed to objectives to lay stress on the favourite criteria. The choice of relative weights used for the different composite objectives was quite arbitrary–the approach was very over-fitted. The authors pointed out the difficulty to deal with multiple the objective nature of the problem using the weighted sum approach.

In our best of knowledge, Falkenauer (Falkenauer, 1995) proposed a first genetic algorithm for ALB with 'resource dependant tasks times' algorithm based on a GGA and a B&B algorithm. The method was able to supply a well balanced and cheap assembly line. The minimal and maximal number of stations was computed by solving the classical ALBP with respectively the fastest and slower resource assigned to all tasks. The GGA distributed the tasks onto stations, while the B&B algorithm selected the optimal resource for each station. The main advantage of this method

was its remarkable computation speed for this NP-Hard problem, even for a large number of operations.

In the next section we will present our method to treat the resource planning for assembly line problem. We will introduce a multiple objective grouping genetic algorithm (MO-GGA), based on the 'equal piles' approach. The concern is the quality of the resulting line in terms of balancing, cost, reliability, etc.

3. Dealing with real-world hybrid assembly line design

The main goal is to assign tasks to stations and to allot resources to tasks. This would lead to a design of the assembly line by considering criteria such as the cost of the line, its availability and its balancing.

> *A task in the point of view the RP is the combination of feeding, handling and insertion. A set of possible groups of equipment (feeders, handlers, insertion devices) are called "functional groups" (FG). For more details about this philosophy, the reader is referred to* (Pellichero, 1999).
>
> **In the remainder of this chapter a set of FG are associated to each "task" or "operation" and the term "equipment" means a set of elementary[3] equipments.**

The hybrid assembly line design problem can formulated as follows. Given a set of tasks, and for each task a set of possible resources each characterised by its price, reliability, stations space and speed in terms of the resulting duration of the task, and given the constraints of cycle time and max peak time[4] (in the case of variants) and, possible, precedence among some tasks, find

- the resources to be allocated to each task, among the possible ones;
- an assignment of tasks to stations along the line, such that

  - no precedence constraint is violated;
  - the station's workload is equal as possible to the cycle time of the line;
  - in the case of multiple products the average process time of each station does not exceed the max peak time.

The following objectives are to be met (not necessarily all together).

- The total price of resources allocated to tasks is minimal.
- A maximal reliability of the line is attained,

---

[3] An elementary equipment can be any feeding, handling or insertion equipment or any auxiliary operation one can find in assembly lines (checking, adjusting, cleaning,...).

[4] This maximum peak time may not be exceeded by any variant process time on a station, while the cycle time must not be overstepped by the average working time on a station.

- The surface occupied by the equipment fit the station space,
- The workload of the stations is as balanced as possible.

A set of features are associated to each task 'functional group', a cost, a process time, an availability, an equipment space and a list of its incompatible functional groups.

3.1. Cost

The cost of a FG will be given by the sum of the costs of each of its pieces of equipment. The price of a resource is its price over the expected lifespan of the line. They will include:

1. the purchase cost;
2. the exploitation cost added to the maintenance cost;
3. the cost of manpower necessary to use the equipment, including possible training, etc.;
4. the consumption cost.

Only the purchase cost is fixed. The three other costs are variable ones. In order to evaluate them correctly, it is thus necessary to compute them for a given period of time (a year for instance).

3.2. Process time

The estimation of the process time of an elementary task is far from being simple. There is still a lack of reliable tools for the estimation of these times, except for the manual feeding, handling and insertion times estimations where the work of (Boothroyd, 1992) has brought a considerable improvement. Thus, further research on this field is still needed. A research work is in progress at the Mechanical Engineering department at the UCL (Université Catholique de Louvain) (L'Eglise, 2000).

Due to the possible hidden times, it is known that the global duration of an assembly operation is not always the sum of the process times of the equipments in the FG. Also, the duration of two successive and distinct FG is not always the sum of their process time. In order to deal with these kinds of exceptions, only an *interactive* constitution of FGs can yield to their correct processing time (Pellichero, 1999). There are two kinds of hidden times: inside a FG and among several FGs.

3.2.1. Hidden time inside a FG

Let us illustrate how we take these parameters into account. Suppose that we have a group consisting in picking up a part with a manipulator and placing it on a hole where a press will insert it, as illustrated at Figure 7.2.

The complete process time of the operation can be subdivided into

- a visible time, i.e., the part of the operation which cannot be executed simultaneously with another one;
- a hidden time, representing ‚parts' of the operation that can be executed in parallel.

In our example, the picking-up of the part by the manipulator from its storage location can be done simultaneously with the insertion of another part. The insertion phase can begin only when the manipulator is picking-up a new part.

	Press	Manipulator
Visible time	3s	6s
Hidden time	0s	2s
Total Time	3s	8s

Table 7.1. Subdivision of the operating times.

If we refer to the data given at (Table 7.1) and illustrated at Figure 7.2, the total visible process time should be the sum of the visible time of the two equipments, i.e., 6s + 3s = 9s. This duration being of 9s is greater than the 8s necessary for the manipulator to accomplish a complete operating cycle. The actual operating time of the group to consider when balancing the line will thus be 9s in our example.

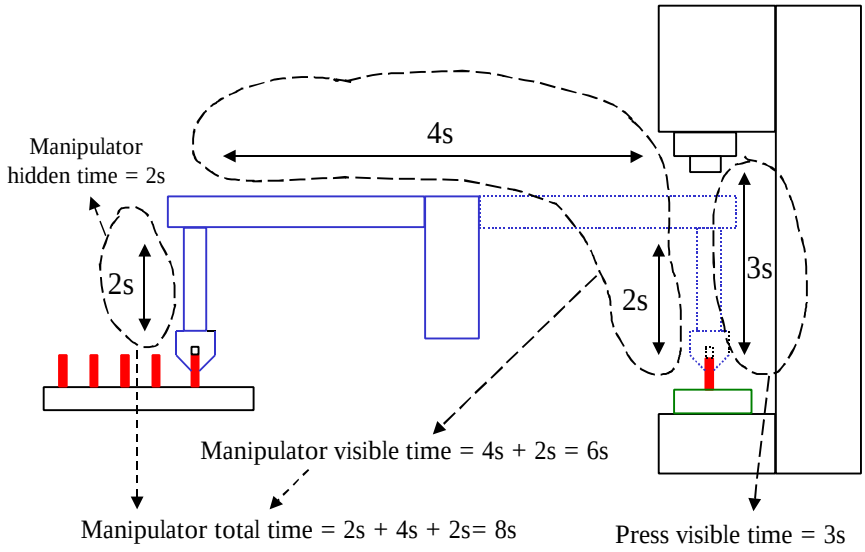

Figure 7.2. Illustration of the process times decomposition (Pellichero, 1999).

The operating time of a given group will thus be given by the sum of the visible times of each equipment or by the largest total operating time of a piece of equipment in the group if it is larger than that sum (Pellichero, 1999).

3.2.2. Hidden times between FGs

Hidden time between FGs is the second kind of hidden time that can exist between tasks. For instance, let us consider the case illustrated in Figure 7.3. It shows two different FGs (FG_1 and FG_2) coupled on the same station which could simultaneously execute their corresponding operations. The process time of the station will therefore not be equal to the sum of the process time of the FGs, but rather to the larger process time of the two FGs.

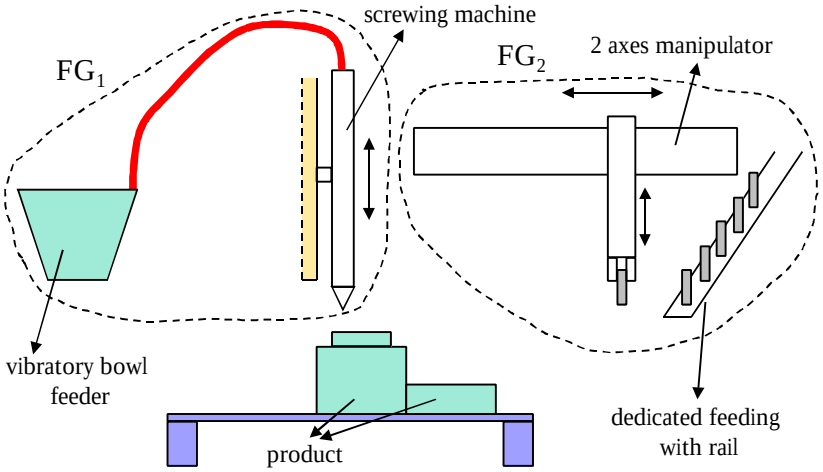

Figure 7.3. The coupling of FGs on the same station (Pellichero, 1999).

The designer usually uses the G*antt diagram*, as illustrated in Figure 7.4 to estimate the resulting process time. Indeed, in this example, FG_2 starts 2s after the start of FG_1 i.e. *1s* before the end of the process cycle of FG_1. It seems therefore obvious that the exact process time of the two FGs cannot be automatically pre-determined, using simple rules, as it was the case for the first kind of hidden times. The process times of combinations of FGs have thus to be pre-defined by designers. For this purpose, an *exception list* is associated to each FG. Each element of this list is composed by a set of FGs and their correspondent process time. Figure 7.4 shows an example where the exception list is {(FG_1, FG_2), 6}. Thus, if a station contains FG_1 and FG_2 the process time is set to *6* and not the sum of their process time which is *8 (2+6)*.

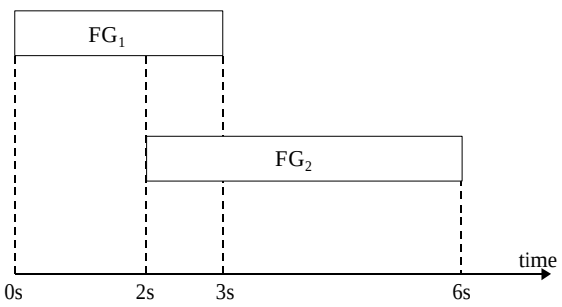

Figure 7.4. An example of Gantt diagram.

3.3. Availability

The availability of a piece of equipment is defined as the proportion of total time that it is available for use. Therefore, the availability of a repairable piece of equipment is a function of its failure rate and of its repair or replacement rate (O'Connor, 1995).

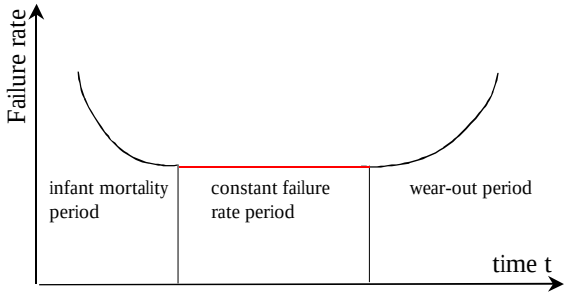

Figure 7.5. Failure rate as a function of time.

Figure 7.5 shows the evolution of the failure rate during the lifetime of an equipment. It is traditionally defined by the so-called *bath-tub curve* (Rampersad, 1994). It is clear that the failure rate depends on the lifetime *t*. Three periods can be distinguished from this curve:

1. the *infant mortality period*, in which the failure rate drops;
2. the *constant failure rate period*, in which accidental failures occur;
3. the *wear-out period*, in which the failure rate rises due to technical age.

Assuming a *constant* failure rate is equivalent to consider that the equipment is in its *normal* functioning period, corresponding to the vast majority of its lifetime. The availability of the FG will be computed by a combination of the availabilities of its pieces of equipment. It will generally be the *product* of the availabilities of the pieces of equipment belonging to the FG as they have a *serial* configuration.

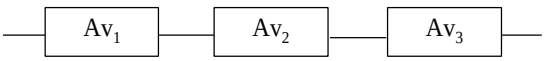

Figure 7.6. Representation of a FG as a serial system.

When making this assumption, the pieces of equipment of a functional group can be represented by a serial system, as illustrated at Figure 7.6. The availability of the FG is equal to $Av_1 * Av_2 * Av_3$. Thus, when several FGs are grouped on the same station, the availability of the station will be given by the product of availabilities of the FGs.

The dependence between these equipments was neglected. This assumption can correspond to the large majority of the cases, but we are aware of its limitations. Thus, further research on the way to model such systems will be largely welcome.

3.4. Stations space

Station space is proportional to the space for storage of parts used on the station as well as the space occupied by the equipments. The storage space (Storage_Space) is the space needed to store parts before being assembled. A constant C_i is assigned to each equipment, it is proportional to the space it occupy. Figure 7.7 shows a simple representation of the space occupied by a set of equipments. Suppose that the station is composed of *k* equipment units. The station space (Station_Space) is given by:

$$Station\_Space = (\sum_{i=1}^{k} C_i) + Storage\_Space$$

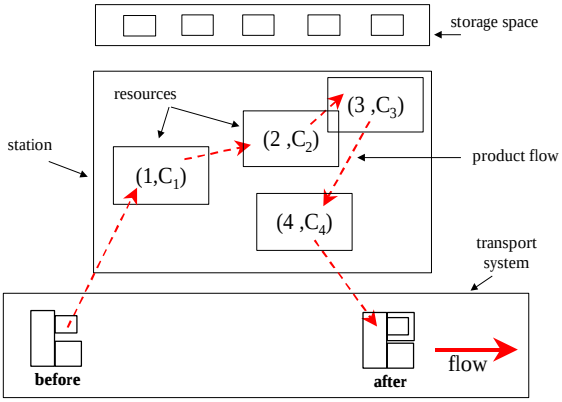

Figure 7.7. Station space.

This formulation is far from being realistic. Nevertheless, it can help to avoid crowded stations. In order to really estimate the space occupied by a set of equipments, more information on the shape of the equipment as well as the shape of the station are needed. In this case, another famous problem arises which is the 2D-Bin Packing 'problem. This leads us to the physical layout problem which is one step more in the design of assembly line. An interaction between the two assembly line design modules namely logical and physical layout is really needed.

3.5. Incompatibilities among several types of equipments

Another element we should take into account is the possible incompatibility between certain kinds of equipments. Indeed, it can happen that two categories of equipments cannot be grouped on the same station. This is considered as a *hard* constraint to comply with, and not an optimisation criterion. A *matrix* of incompatibilities called *I* (see Figure 7.8), is introduced to deal with this problem. The size of the symmetric matrix *I* is equal to *N*N* where *N* is the number of equipments and each I_{ij} element of that matrix will have the value:

- I_{ij} = 0 if equipment *i* is compatible with equipment *j*,
- I_{ij} = 1 if equipment *i* is not compatible with equipment *j*.

$$\begin{array}{c} \\ Eq_1 \\ Eq_2 \\ Eq_3 \\ \vdots \\ Eq_N \end{array} \begin{array}{c} \begin{array}{ccccc} Eq_1 & Eq_2 & Eq_3 & \dots & Eq_N \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & \dots & 1 \\ 0 & 1 & 1 & \dots & 1 \\ 0 & 1 & 1 & \dots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 0 & \dots & 1 \end{pmatrix} \end{array}$$

Figure 7.8. Matrix of incompatibilities between equipments.

4. Input data

In brief, we need at least the following data to design an HAL (Figure 7.9):

- the desired number of stations;
- the desired cycle time;
- the precedence constraints between operations;
- the user's preferences for the operation mode (manual, automated and/or robotic);
- the list of exceptions to deal with hidden times;
- an equipment database which yields the features of the different resources (cost, reliability, process time, equipment space);
- the list of its incompatible equipments.

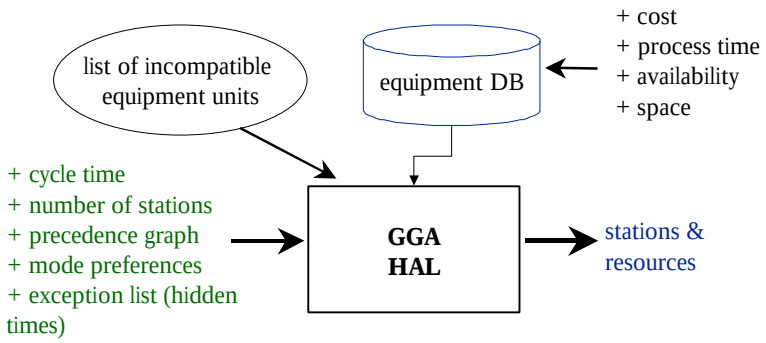

Figure 7.9. Data flow for hybrid assembly line.

The preparation of data and especially the 'equipment selection' step yields to a set of equipments which can perform the given set of operations. Economic criteria are used in the evaluation of the equipment selection process (Pellichero, 1999).

5. Overall method

The line balance efficiency is impacted by the number of stations in the line and the idle time. Normally, the less the number of stations and the less the idle time is, the more efficient the line is. One of our aims is to minimise the whole cost of the line. Given the fact that faster resources are more expensive (and cheaper resources are slower), the cheapest line can present fast and slow equipment together and can feature a small or a high number of stations. Thus, we have to decide which task will be performed on which station but we also must select the resource allocated to each of them. There is an important link between the two stages. We propose the following algorithm to generate possible solutions of the problem. The individual construction algorithm (ICA) is used each time we have to construct a solution for a problem.

ICA algorithm:

1) *Assign tasks to the stations (using the process time corresponding to the fastest equipment) according to an equal piles strategy;*
2) *Generate all possible resource combinations for each station thanks to a branch & cut algorithm;*
3) *Select the best equipment combination for each station using the PROMETHEE II decision-aid method.*

Since our aim is to deal with many conflicting objectives which consist in maximising the balancing, minimising the cost, maximising the reliability, minimising the station space, we settled for the MO-GGA approach introduced in Chapter 5. The main steps are the following (Figure 7.10):

MO-GGA algorithm:

1) Create a population of individuals using the individual construction algorithm (ICA).
2) Use the decision-aid method PROMETHEE II to order individuals in the population.
3) Recombine (mate) best individuals (parents) to produce children (using ICA).
4) Mutate children.
5) Use PROMETHEE II to order the new population.
6) Replace the worst individuals of the population by the new children.
7) If a satisfactory solution has been found stop. Else go to 3).

The initial population is generated using ICA. The individuals are then ranked using the PROMETHEE II method. At each iteration of the main loop, the better solutions are selected from the current population. Recombination produces a number of new individuals (offspring). The mutation is used to explore the search space. The offspring are then incorporated into the original population. Again, the individuals are ranked using the MCDA. The loop finishes when the termination criteria given by the user (e.g., convergence, maximum number of generations, etc.) are reached.

The PROMETHEE II method permits to maximise a set of objectives and minimise another set simultaneously. For each objective a triplet (*p*, *q*, *w*) is introduced, where *w* is the weight, the values *p* and *q* are the preference and the indifference thresholds respectively (see Appendix 3). The choice of one solution over the others requires problem knowledge. It is the DM task to adjust the *weights* to help the algorithm to find good solutions. Optimising a combination of objectives has the advantage of producing a single solution, requiring no further interaction with the DM. For given user's preferences and a given design problem we run the following multiple objective GA. The basic features of the MO-GA are illustrated in the following pseudo-code.

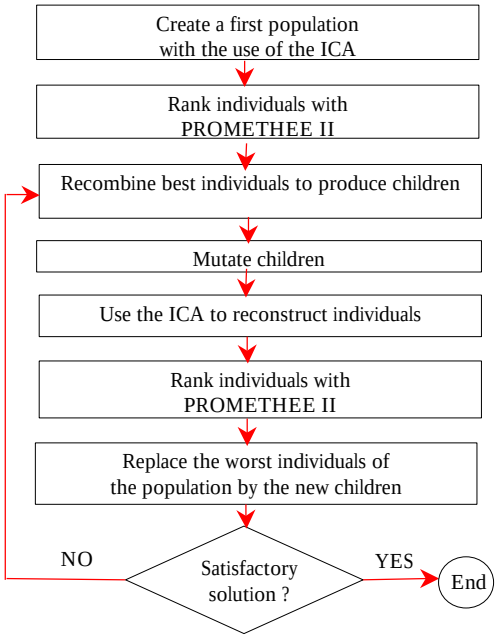

Figure 7.10. Steps of the MO-GA.

5.1. Distributing tasks among stations

A solution to a given instance of the problem is a set of tasks with their respective resources in a set of stations. Dealing with hybrid assembly line, introduce an additive constraint which is the kind of tasks to be grouped. In this section we recall the strategy used to group tasks and introduce the operating mode as a new constraint.

5.1.1. Equal piles Algorithm

In order to assign operations to stations, we use our EPAL (equal piles in assembly line) introduced in Chapter 6. The hard constraint is the fixed number of stations (piles). At the operations-to-stations stage, we use a minimum cycle time *min_ct*. This *min_ct* is the ratio between the sum of minimum process times of the operations and

the desired number of stations. The approach to solve the problem is based on the so-called 'boundary-stones'(see Chapter 6 section 3).

5.1.2. Mode preferences

The feeding, handling and insertion method of each part of the product to be assembled, yield associative and dissociative preference constraints. The associative constraints can be defined by the following sentence: *"The manual tasks have to be grouped together, and the robotic or automated tasks have to be grouped together"*. The dissociative preference constraints can be defined by: *"The manual tasks cannot be grouped with robotic or automated tasks"*.

The resulting preference constraints impeach manual operations to be grouped with robotic or automated ones (grouping constraints). Each operation has one or several possible modes according to the equipment which is able to perform it. Note that if several modes are allowed for an operation, the mode will be fixed by the *optimisation module* to yield the *best* logical layout. Those constraints are said to be hard, because they cannot be violated. Dealing with these constraints lead us to propose more realistic results of the logical line layout. Chapter 10 gives more details about the influence of mode preferences of tasks on the design of assembly line.

5.2. Selecting equipments

The equipment selection problem is an essential part of manufacturing system design. It typically involves the careful choice of a set of equipments to be used in production based on known and often predictable technical and economical criteria. The technique developed helps the GGA to select the appropriate set of equipments to be used. The choice is guided by the objective fixed by the designer.

This selection step involves selecting the type (operating mode) and the list of equipments required to perform a variety of operations. Thus, each time a set of tasks has been assigned to a station the process time, cost, reliability and stations space has to be computed. This problem is not easy, because there can be several resources available for each task assigned to a station. Each choice of resources imply a different cost and duration of tasks.

Clearly, this is a combinatorial problem: in principle, any assignment of resources to the tasks of the given station could be interesting, and there is an exponential number of these assignments. Fortunately, due to the cycle time constraint the resources must be fast enough to perform all the tasks in that timeframe. This allows us to define a B&C procedure that efficiently explores the 'huge'search space.

The important point is that the B&C procedure is applied to each station separately. Consequently, even for large problem instances, the B&C typically handles only a small number of tasks at any given time, because even with the fastest resources it is

usually impossible to assign a large number of tasks to a station in realistic situations. The size of the B&C problem thus stays reasonably small, leading to an acceptable speed of the method.

The choice of a best solution is far from being easy. The multi-criteria decision-aid method PROMETHEE II is thus used. It helps to deal with the different objectives addressed by the designer.

The main features of the algorithm can be summarised in the following steps:

develop the first node (task) level i=0;
verify the validity of the offspring nodes;
repeat
if there is no valid nodes then stop;
generate all the offspring(s) of all valid nodes : create level i+1;
verify the validity of the offspring's nodes of level i+1;
until *the last level is attained;*
use PROMETHEE II to choose the best solution among the valid ones.

5.2.1. Branch and cut algorithm

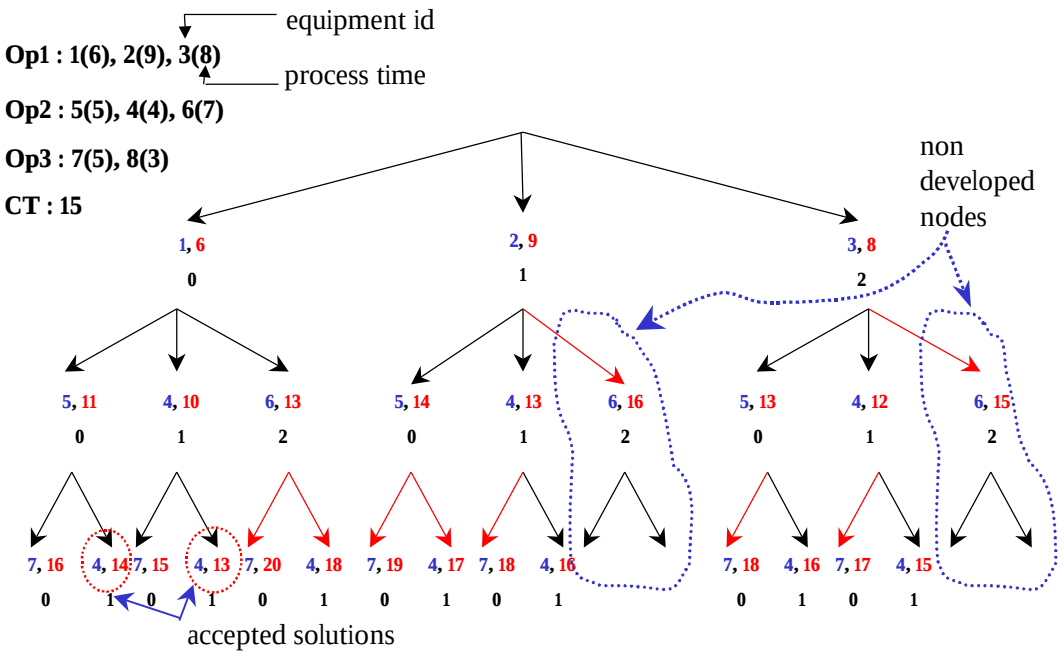

Figure 7.11. Tree generated by the branch and cut algorithm.

As its name implies, the branch & cut method consists in two fundamental procedures: branching and cutting. The search procedure may be represented as a tree, the root symbolising the whole problem. Branching is used to divide a large

problem into two or more sub-problems usually mutually exclusive. A branch is associated to each sub-problem. These can be partitioned in a similar way, yielding new branches and so on. Cutting permits also to stop partitioning non valid sub-nodes. The associated branches are not further developed. The partitioning process stops if it represents only one solution, or if it can be shown that the node does not contain an optimal solution.

The branch & cut algorithm is used to assign equipments to operations. Initially, the tree contains only one node, the first task (level *0*). The branching is done by developing a set of possible resources for the given task. Each node corresponds to an equipment and each level to one task. On the graphic presented at Figure 7.11, each couple *(a, b)* corresponds to an equipment, and the sum of process times of this branch at the given level. For example, the couple *(5, 11)* means that by selecting equipment *5* to realise operation *2*, the total process time of station is *11*. At each level, all possible equipments of the given operation are generated but only the valid branches respecting the constraints of the problem (cycle time, compatibility, etc.) are developed further. For instance, selection of equipment *2* for operation *1* and equipment *6* for operation *2* yields an process time of *16*. Since the cycle time is set to *15*, this branch will not be developed further. Once all the levels (valid branch) have been developed, only valid solutions are kept. By valid solutions we mean the solutions verifying the following constraints:

- the sum of process time of operations for the selected equipment must not exceed the cycle time.
- the list of equipments used at each station must not be incompatible.

Suppose we have n tasks and each task i has b_i possible pieces of equipments. The number of levels equals n which is the number of tasks, while the number of nodes is given by:

$$NbNods = \sum_{j=1}^{n} (\prod_{i=0}^{j} b_i) * b_j \qquad with \quad b_0 = 1;$$

The number of end-leafs is equal to :

$$NbEndLeaf = \prod_{i=0}^{n} b_i$$

For instance, in case $n=3$ and $b_i = 2$ for $i=\{0, 1, 2\}$, the number of nodes is *14*, and the number of end-leafs is *8*. It is impossible to *enumerate all feasible* solutions, indeed, their number increase exponentially with the problem size. The aim is to design an algorithm which is faster than enumeration techniques. We are not talking about *bounding* because all the valid nodes of a given level are developed at the same time. Also due the constraints of the problem it is quite difficult to estimate the *lower/upper* bound of the tree. The *cutting* mechanism is used to save time while exploring the

tree. Indeed, it is very interesting to stop the branching of a non valid branch, if any knowledge can help us to know it in advance.
The following decisions have a great influence on the run time of the cutting algorithm: (1) how the tasks are assigned to the different levels? (2) how are ordered the equipment of each task? (3) since there is a set of tests of validity of the solution to be done which one to begin with?. Many tests were done as well as deep thought to save time in useless tests.

5.2.2. How and where does the user intervene?

We present here the criteria used during the selection of equipment.

- Process time: each station should not require more than a cycle time to perform all the tasks.
- Cost: the total price of the resources allocated to the stations must be minimised.
- Reliability: must be maximised on each station.
- Space: is proportional to the space occupied by the station, and must be reduced.

Thus, once all the end-leafs of the tree have been found, time come to choose the best solution among the valid ones. Since each solution is characterised by its cost, process time, reliability, etc. classical *pairwise* methods to compare solutions cannot be used. The different solutions found by the B&C algorithm serve as an input data for the PROMETHEE II method to choose the best equipment taking into account the different criteria. Afterwards, resources are assigned to each task of the given station.

A non valid solution can be:

- a solution where the sum of process times of the fastest equipment exceeds the cycle time,
- solutions composed by only incompatible equipments,
- the desired cycle time is incompatible with the fixed number of stations,
- too many tasks are grouped on a given station.

If there is no possible (valid) solution among the developed nodes a solution corresponding to the fastest equipment is then selected.

5.3. Heuristics

Two new heuristics are introduced to deal with the hard constraint of operating mode of the tasks. Indeed, as pointed out in section (4.1.2.) the manual operations cannot be grouped with robotic or automated operations. Violating this constraint will yield a non valid hybrid assembly line. Imagine a set of manual tasks executed by an operator grouped in the same station with other tasks executed by a robot. It is clear that the risk that the operator will end at the hospital is quite high. Now, we

know that the constraint is a hard one and we have to take it into account, still to find a way to deal with.

We propose two heuristics to be alternatively used to improve the solutions obtained by the boundary stones algorithm: the merge and split heuristic and the pressure difference heuristic. Both of them will be executed on a solution until no improvement is obtained anymore, or a maximum number of trials is reached.

Merge and split

Figure 7.12 (a) represents a kind of situation one has to face when dealing with the operating mode of hybrid assembly line. Suppose there are two non-filled adjacent manual stations and an over-filled automated one. In order to find a good balancing, one way is to merge the two manual stations and to split the automated one. Indeed, since the hard constraint is the fixed number of stations, merging two stations obliges to split another one.

Figure 7.12 (b) represents the solution obtained after the merge and split procedure. The result is one manual station and two automated stations. In the balancing point of view, the second solution is better than the first one. Note that the sum of the process time of the two new automated stations (70% and 90%) is not necessarily the process time of origin (130%). Indeed, the equipment assignment algorithm may choose other equipments to the given tasks.

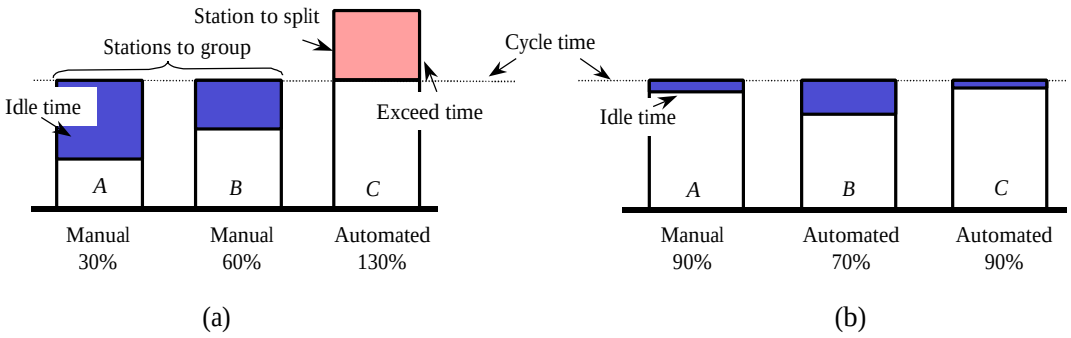

Figure 7.12. Solution before (a) and after (b) the merge and split heuristic.

Pressure difference

The main idea behind the heuristic gave it its name. It begins by finding a station exceeding the cycle time (the high pressure) as well as the station less filled (less pressure). The goal is to move the exceeding process time of station *C* in Figure 7.13(a) to fill the gap (idle time) existing on station *A*. The operating mode and the precedence constraints of the tasks to move have to be verified. In this case a task *i* to move from station *C* must have all its predecessors in station *A* (or before). If the move had been from *A* to *C*, all the successors of task i would have to be in *C* or later.

Figure 7.13(b) represents the solution obtained after executing the procedure. The kind as well as the number of stations obtained is the same before the application of the heuristic. The simple wheel and multiple wheel heuristics[5] cannot improve such a solution, since the two manual stations are separated by an automated one. It is clear that the second solution is better balanced than the first one. Note that the operating modes, the equipment compatibility and the precedence constraints of each task has to be verified each time a move is made.

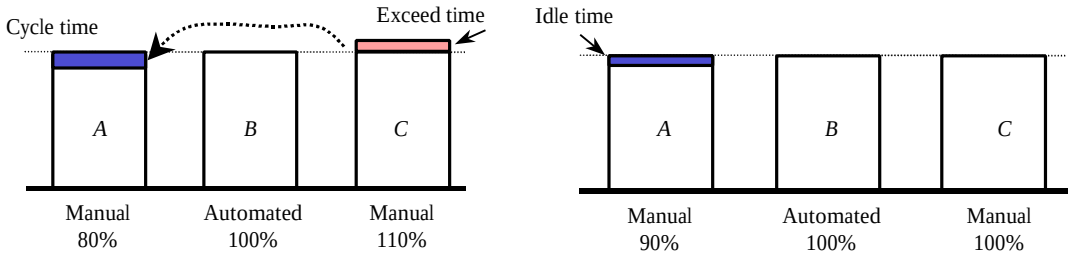

Figure 7.13. Solution before (a) and after (b) the pressure difference heuristic.

5.4. Dealing with multi-product assembly line

In this section, the method used to deal with multi-product resource planning is developed. The main goal is to find the cheapest assembly system. The difference between the multi-product method and the single product RP is the way we compute the process time of a given station. The approach is the same for the two problems.

A *maximum peak time* parameter fixed by the designer, is introduced to allow some variants' process times to exceed the desired cycle time. This maximum peak time may not be exceeded by any variant process time of any station, while the desired cycle time must not be overstepped by the average working time on any station. The classical case is the one where the maximum peak time is equal the cycle time–which corresponds to the single-product RP.

Initially, the tree contains only one node, the first task (level *0*). The branching is done by developing a set of possible resources for the given task. Each node corresponds to an equipment and each level to one task. On the graphic presented at Figure 7.14, in each box, the couple *(a, b)* corresponds to an equipment, and the average process time among all variants in that branch at this level. For example, the couple *(2, 4.5)* at level *1* means that by selecting equipment *2* to realise operation *1*, the average process time on the station is *4.5*.

At each level, all possible equipments of the given operation are generated but only the valid branches respecting the constraints of the problem (cycle time, maximum

[5] The simple and multiple wheel heuristics were introduced in the EPAL algorithm (see chapter 6).

peak time, compatibility, etc.) are developed further. For instance, selecting equipment *1* for operation *1* and equipment *6* for operation *2* yields an average process time of *9.5* and a process time of 13 on variant 1 and a process time of 6 on variant 2. Even if the average process time (*9.5*) is less than the cycle time (*10*), since the process time of variant (*13*) exceeds the maximum peak time, this branch is not valid. Using equipment 1 for operation 1 and equipment 4 or 5 for operation 2 leads to a valid solution. In order to select the best set of resources, once again the PROMETHEE II method will be used. By valid solutions we mean the solutions verifying the following constraints.

- the average process time of operations of the selected equipments must not exceed (on average) the cycle time.
- the process time of operations of any of the *variants* for the selected equipments must not exceed the 'maximum peak time'.
- the equipment used at each station must not be incompatible.

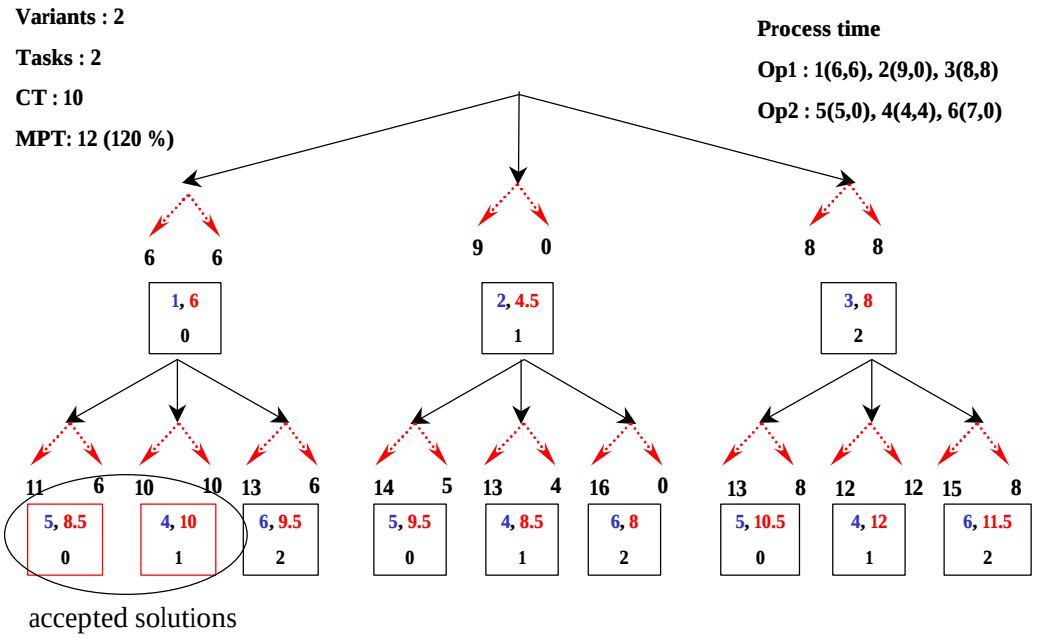

Figure 7.14. Multi-products tree generated by the branch & cut algorithm.

5.5. Complying with hard constraints

We will not use the penalty approach used in most methods, in fact the GGA developed deals only with valid solutions. In the proposed algorithm and tending to deal with real-world applications, i.e., constraints and preferences introduced in section 3, solutions are tested at each level. The method verifies incompatibilities among equipments, updates the process time of the stations using the hidden times, etc. The main features of the method can be summarised in the following steps.

repeat *for each level*
- evaluate the process time of each variant, the average process time of station,
- evaluate cost, reliability, stations space;

  repeat *for all nodes*
  - cut the branch if the used equipment are incompatible;
  - update the process time using the masked time;
    - if the average process time exceeds the cycle time, cut the branch;
    - if the process time of a given variant exceeds the maximum process time, cut the branch;

  until *the last node is attained;*

until *the last level is attained.*

6. Application of the method

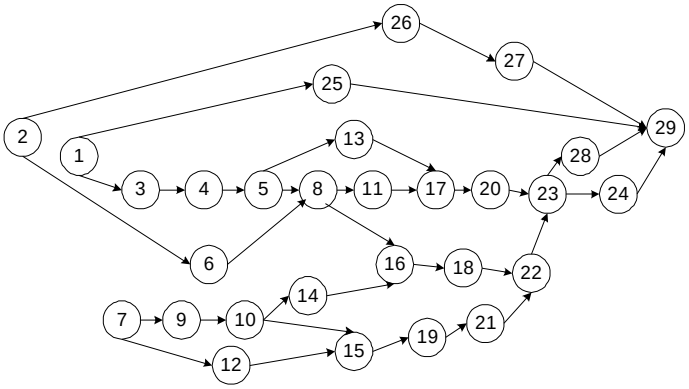

Figure 7.15. Precedence graph of the problem.

The case study we propose is adapted from a problem proposed in the line balancing benchmark suite[6] of (Scholl, 1999). The benchmark presented here is called 'BUXEY'. It considers 29 tasks with precedence constraints illustrated at Figure 7.15. We proposed three possible equipment (and operating times) for each operation. We considered that equipments had the same reliability (99 %) and same space factor (1).

We have tested our algorithm for several numbers of stations (N) and several desired cycle times (C). The results of the GGA are presented at Table 7.2. It shows the total cost of the line (arbitrary units), and the loads of stations (ratio of the sum of process times and the cycle time). Note that the station load is superior to 1 in some cases, meaning that the desired cycle time cannot be held for the selected number of stations. As it can be seen, the line will generally be less expensive as the cycle time

[6] The benchmark suite can be accessed via the Web at
http://www.bwl.tu-darmstadt.de/bwl3/forsch/projekte/alb/index.htm

constraint is relaxed (cycle time augments). But this is not always the case. For example, for seven stations, the cost raises as the cycle time augments. This is because the cost is not the only criterion taken into account. The line is better balanced for higher cycle times, for slight differences of the line cost.

N, C	Cost	Stations loads
6, 44	3340	1.00, 1.00, 1.05, 1.07, 1.05, 1.00
6, 45	3340	1.00, 1.00, 1.00, 1.00, 1.00, 1.00
6, 46	3280	1.00, 1.00, 1.00, 1.00, 1.00, 1.00
7, 38	3230	1.00, 1.05, 1.03, 0.97, 1.05, 1.08, 1.00
7, 39	3240	1.00, 1.03, 1.03, 1.00, 1.03, 1.00, 1.00
7, 40	3270	1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00
8, 34	3280	1.00, 1.00, 1.03, 1.00, 1.03, 1.00, 1.00, 1.00
8, 35	3240	1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00
8, 36	3030	1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00

Table 7.2: Results of the HAL balancing algorithm.

7. Conclusions and further works

This chapter presents a new method to address the hybrid ALB problem with multiple objectives. The aim is to assign tasks-to-stations and select the assembly equipment to perform each of them. The goal is to minimise the total cost of the line by integrating design (stations space, machine real cost, etc.) and operation issues (cycle time, precedence constraints and availability, etc.). We used the MO-GGA to tackle the problem, hybridised with a B&C and the multi-criteria decision-aid method PROMETHEE II. This method is integrated in the software package SELEQ (SELection of EQuipment) (Pellichero, 1999), which is a user-friendly tool to design assembly line. Special user's preferences are taken into account by the proposed method. The essential concepts adopted by the method are described.

The architecture of the proposed model shows how it is easy to incorporate industrial features in the algorithm. We believe that we are not far from real-world problems, but we still need more tests and interaction with industrials. Once again, this is just a prototype, there still more work on this field.

8. References

(Bard, 1989) Bard J.F., 'Assembly line balancing with parallel stations and dead time', Int. J. Prod. Res., Vol. 27(6), pp. 1005-1018, 1989.

(Baybars, 1986) Baybars I., 'A survey of exact algorithms for the simple assembly line balancing', Management Science, Vol. 32, 909-932, 1986.

(Boothroyd, 1992) Boothroyd G. 'Assembly automation and product design', Marcel Decker Inc, 1991.

(Brans, 1994) Brans J.-P., Mareschal B., 'The PROMCALC & GAIA decision support system for multicriteria decision aid', Decision Support Systems, North-Holland, Vol. 12, pp. 297-310, 1994.

(Bukchin, 2000) Bukchin J. and Tzur M., 'Design of flexible assembly line to minimise equipment cost', to appear in IIE Transactions, 2000.

(Faaland, 1992) Faaland B.H., Klastorin T.D., Schmitt T.G. and Shtub A., 'Assembly line balancing with resource dependent task times', Decision Sciences, Vol. 23, pp. 343-364, 1992.

(Falkenauer, 1995) Falkenauer E. and Delchambre A., 'Integrated assembly and resource planning in production line design', Proceedings of ISATP'95, Pittsburgh, Pennsylvania, pp. 220-225, 1995.

(Gaither, 1996) Gaither N., 'Production and operations management', Belmon, CA: Duxbury, 1996.

(Graves, 1979) Graves S.C. and Withney D.E., 'A mathematical programming procedure for equipment selection and system evaluation in programmable assembly', Proceedings of the IEEE Decision and Control, pp. 531-536, 1979.

(Graves, 1983) Graves S.C. and Lamar B.W., 'An integer programming procedure for assembly system design problems', Operations Research, Vol. 31(3), pp. 522-545, 1983.

(Graves, 1988) Graves S.C. and Holmes Redfield C., 'Equipment selection and task assignment for multiproduct assembly system design', International Journal of Flexible Manufacturing Systems, Vol. 1, pp. 31-50, 1988.

(Gustavson, 1986) Gustavson R.E., 'Design of cost-effective assembly systems', C.S. Draper Laboratory Report, N°. P-2661, Cambridge, 1986.

(Gutjahr, 1964) Gutjahr A.L. and Nemhauser N.K., 'An algorithm for the line balancing problem', Management Science, Vol. 11(2), pp. 308-315, 1964.

(Helgeson, 1961) Helgeson W.B. and Birnie D.P., 'Assembly line balancing using the ranked positional weight technique', J. Ind. Engng, Vol. 12(6), pp. 394-398, 1961.

(Lee, 1991) Lee H.F. and Johnson R.V., 'A line-balancing strategy for designing flexible assembly systems', International Journal of Flexible Manufacturing Systems, Vol. 3, pp. 91-120, 1991.

(Lee, 1996) Lee H.F. and Stecke K.E., 'An integrated design support method for flexible assembly systems', Journal of Manufacturing Systems, Vol. 15(1), pp. 13-32, 1996.

(L'Eglise, 2000) L'Eglise T., Marée J-F. and Raucent B., 'A complete methodology for generating an assembly system', Proceedings of FAIM'2000, Maryland, USA, pp. 823-832, 2000.

(Malakooti, 1996) Malakooti B. and Kumar A., 'A knowledge-based system for solving multi-objective assembly line-balancing problems', Int. J. Prod. Res., Vol. 34(9), pp. 2533-2552, 1996.

(McMullen, 1998) McMullen P.R. and Frazier G.V., 'Using simulated annealing to solve a multiobjective assembly line balancing problem with parallel stations', Int. Jour. Prod. Res., Vol. 36(10), pp. 2717-2741, 1998.

(Minzu, 1997) Minzu V. and Henrioud J.-M., "Approche systématique de structuration en postes des systèmes d'assemblage monoproduits", RAIRO-APII-JESA, Vol. 31(1), pp. 57-78, 1997.

(Nevins, 1989) Nevins J.L. and Withney D.E., "Concurrent design of products and processes", Mc Graw-Hill, New York, 1989.

(O'Connor, 1995) O'Connor P.D.T., "Practical reliability engineering", Third edition revised, John Wiley & Sons, 1995.

(Okano, 1993) Okano A., "Computer-aided assembly process planning with resource assignment", Proceedings of the International IEEE Conf. on Robotics and Automation, Atlanta-Georgia, pp. 301-306, 1993.

(Pellichero, 1999) Pellichero F., "Computer-aided choice of assembly methods and selection of equipment in production line design". Ph.D. Thesis, Université Libre de Bruxelles, 1999.

(Petit, 1999) Petit F., "Interactive design of a product and its assembly system", Ph.D. Thesis, Université Catholique de Louvain, 1999.

(Pinnoi, 1998) Pinnoi A. and Wilhelm W.E., "Assembly system design: a branch and cut approach", Management science, Vol. 44, No 1, pp.103-118, 1998.

(Pinto, 1983) Pinto P.A, Dannenbring D.G and Khumalawa B.M., "Assembly line balancing with processing alternatives: an application", Management Science, Vol. 29(7), pp. 817-830, 1983.

(Rampersad, 1994) Rampersad H.K., "Integrated and simultaneous design for robotic assembly", John Wiley & Sons, London, 1994.

(Rubinovitz, 1993) Rubinovitz J. and Bukchin J., "RALB–a heuristic algorithm for design and balancing of robotic assembly lines", Annals of the CIRP, Vol. 42, pp. 497-500, 1993.

(Scholl, 1999) Scholl A., "Balancing and sequencing of assembly lines, Heidelberg: Physica, 1999.

(Tsai, 1993) Tsai D.M. and Yao M.J., "A line-balanced-base capacity planning procedure for series-type robotic assembly line", Int. J. of Prod. Res., Vol. 31, pp. 1901-1920, 1993.

(Wee, 1982) Wee T.S. and Magazine M.J., "Assembly line balancing as generalized bin packing", Operations Research letters, Vol. 1, pp. 56-58, 1982.