Worker and floater time allocation in a mixed-model assembly line

Manfred Gronalt and Richard F. Hartl

University of Vienna,

Department of Business Studies,

Brünner Straße 72, A-1210 Vienna,

April 19, 2000

Abstract

Short term workforce planning of labor-intensive transfer lines is addressed in this paper. Although the products (e.g. trucks) are assembled on a serial line, they show large differences in the processing times at each station. These differences are due to the assembling options provided to the customers. Given the assignment of operations to stations, we have to find a loading sequence for the products, a worker allocation and a floater time allocation in order to minimize the whole labour costs at the line. We develop a solution approach for this problem considering a rolling planning horizon. Numerical results show that relevant savings can be achieved.

1 Introduction and problem description

In this paper we propose a simultaneous approach for worker and floater time allocation in a mixed-model assembly line. A mixed-model assembly line is a type of production line where a variety of products similar in product characteristics are assembled. The line considered here corresponds to a moving line with fixed and identical cycle time per station. The products being assembled on that line (e.g. medium size trucks) differ widely with respect to their processing times. Examples of such products could be different models of a truck, where each model may come with a variety of options. Usually, line balancing and sequencing procedures are applied in this environment to reduce the variability of processing times. But the influence of these may be restricted due to technological constraints.

Since per type of truck only a very small number is produced per shift (often only one) it is not reasonable to change the set-up of the line after each type. Thus we are indeed facing a mixed-model assembly rahter than a multi-model assembly line.

The motivation and structural constraints were provided by a local truck manufacturer which will expand its capacities during the next few years. For this it would be most desirable to have a decision support tool at hand for determining the workforce level and the allocation of workers and floaters at the line in order to obtain solutions with minimum total wage costs. Be doing this we are faced with the following tradeoff situation: A constant workforce level at each station leads to discrepancies in the individual workload of workers due to the variety of the processing times. We can 'overstaff' some critical stations which leads to an imbalance of worker utilization. On the other hand, we can fix the minimum worker level for each station and use floaters for smoothing capacity peaks. Additionally to this we

may allow workers to move with their products in order to complete the current operations.

The analyzed assembly line consists of several assembly stations for mainly labor-intensive operations. We can assume that each worker is eligible to be assigned to every station although in real world workers have specific skills. For instance, electricians can perform any electrical operation but no assembly of mechanical parts. Workers allocated to a particular station should stay there for the reminder of the shift. In contrast, floaters are allocated temporarily to a particular station. Moreover, floaters are multi skilled and they earn higher wages than regular workers.

The short term planning problems in mixed model assembly lines are work force allocation and production sequencing (see, e.g. Decker[1993] and Bard et al. [1992]). Bard et al. [1992] present a unified framework for dealing with the production sequencing problem. Decker compares floater usage with buffer planning in her work. She assumes that by using buffers in an assembly line the production sequence can be changed at the line in order to smooth the capacity requirements.

Lee and Vairaktarakis [1997] develop a sequencing model for finding the minimum maximum workforce level for a given production period in a rather similar production environment. In their work they assume synchronous worker movement. In a companion paper Vairaktarakis and Winch [1999] consider the problem of worker cross-training in mixed model assembly line, which is an essential assumption in our case too.

Bartholdi and Eisenstein [1996] propose a decentralized approach for workforce control, which assumes asynchronous worker movement. They consider a case, where all items are identical and workers have different skills. In their paper they

show that sequencing workers on the line from the slowest to the fastest will lead to a balanced line with largest possible output.

We will present a comprehensive operational planning procedure which is applicable for the short term day to day work force allocation at the shop floor. Moreover, we develop a hierachical heuristic procedure for this problem. In this procedure the impact of production sequencing and the interactions with floater time allocation is also addressed for smoothing capacity requirements, i.e. the processing time deviations at the respective stations.

In the development of our models we use the following notations. Given is a set of stations, $i = 1, \ldots, I$ and a number of production cycles, $t = 1, \ldots, T$. For a particular production sequence, the workload at station i and cycle t measured in workers needed per cycle p_{it} is known;p_{it} can be any positive real number. The number of worker at station i is denoted as w_i, which is of course an integer.

The reminder of the paper is organized as follows. In Section 2 we define the line caracteristics and outline the basic assumptions of our models. These form a basis for our solution approach which is developed in Section 3. The results of a computational study are shown in Section 4.

2 Modelling framework

2.1 Line characteristics

As in most mixed model assembly lines the conveyance system moves at constant speed where the products are launched in equal time intervals. Each station has been assigned a set of tasks and balancing has been achieved. There is no buffer

between the stations and no parallel stations. At a particular station usually more than one worker can operate. Any imbalance must be smoothed by allocating floaters to the respective station unless this imbalance can be overcome by utilizing overlaps. For a more general discussion of capacity production smoothing in assembly lines see Decker[1993].

2.2 Worker and floater movements

Workers move downstream while performing their tasks on the product. Upon completion, they return upstream to begin work on the next unit. Usually, this is denoted as asynchronous movement and is, for example, very popular in the apparel industry. Floater's movement is considered synchronous as they may change their current station according to the production cycles. Worker and floater transfer times are neglected.

Depending on the nature of the tasks and the physical layout of the facility, stations may be open or closed. In the latter case it is not allowed for adjacent operators to cross each others boundaries. In contrast, open station boundaries can be crossed, but the extent of the crossing may be restricted.

In our manufacturing environment the operators may finish their works in the next downstream station (positive overlap) or they may move upstream one station for early start of assembly (negative overlap). Further, a set of operators may work simultaneously on a single unit and they are permitted to interfere with another set of workers when performing in an adjacent station. In the following we will describe the different cases of worker movement and interfering constraints. Depending on the extent of positive overlap floater time requirements are calculated for allocation in order to reduce these overlaps. We assume that floaters are always available.

In case of nonconflicting overlaps (see Figure 1), the work on truck 2 in station i can be started as soon as truck 1 has been finished. In station i work on truck 3 in station i can be started as soon as truck 2 has been finished in station i

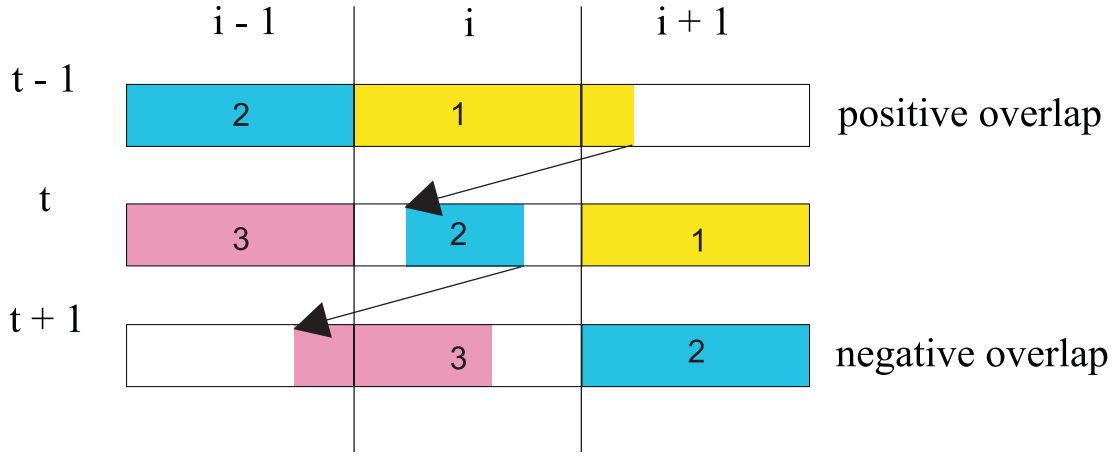

Figure 1: Nonconflicting operations

In contrast, in case of conflicting overlaps (see Figure 2) work on truck of cycle t can start in station i, if trucks of cycle $t-1$ are finished in station i and in station $i-1$

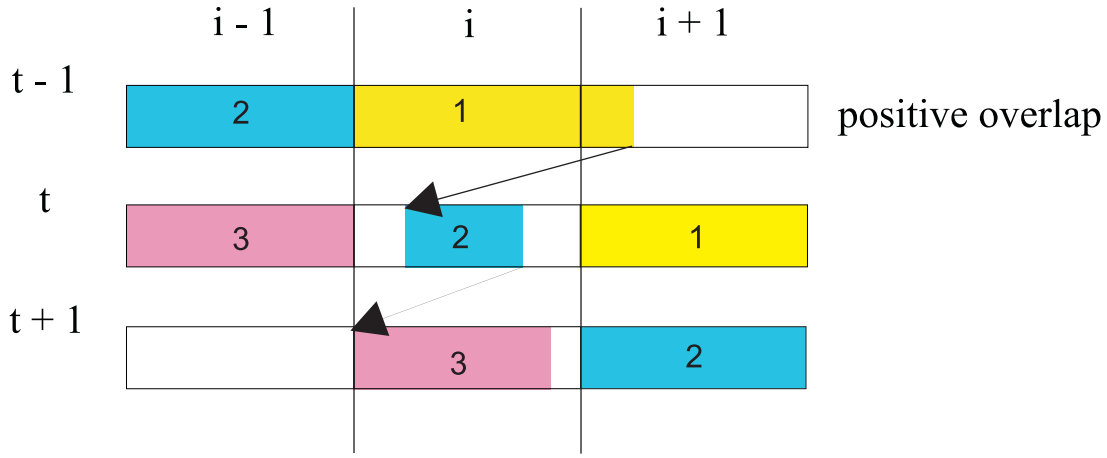

Figure 2: Conflicting operations

3 Heuristic Solution Procedure

Our heuristic approach is a decomposition into four subproblems as depicted in Figure 3. Starting with an initial allocation of workers **(WA)** to the stations an 'optimal' sequencing of the items (trucks) **(TS)** is obtained where the evaluation of the quality of a sequence means solving the Floater Time Allocation problem,**(FTA)**. Thus in the process of improving the sequence the FTA problem has to be solved frequently. This is shown by feedback loop (1) in Figure 3. When different reasonable worker allocations are to be evaluated, each time the TS problem and thus several FTA problems have to be solved. This is indicated by the feedback loop (2) in Figure 3.

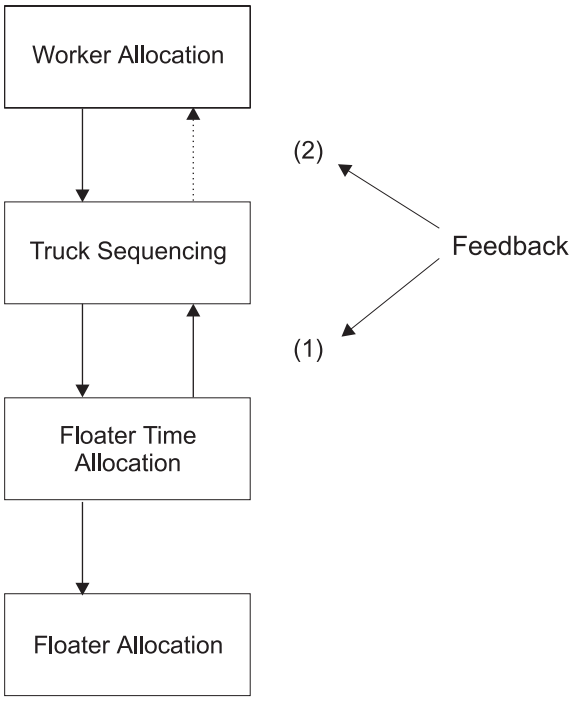

Figure 3: Sequential Planning Procedure with feedback loops

Finally, after a satisfactory solution of the interconnected problems WA, TS and FTA has beed found, the actual working schedule for each floater is obtained by

solving the Floater Allocation problem, **(FA)**. For ease of presentation we discuss the subproblems in the order FTA, FA, TS and WA.

3.1 Floater Time Allocation

We now present various models and solution procedures on how to compute the overlaps and how to allocate the floaters. We first start with the variant with non-conflicting overlaps and afterwards we discuss necessary changes when overlaps are conflicting.

3.1.1 Time model (no floaters)

If no standby workers (floaters) are allocated then there is no decision problem and we can simply compute the overlaps in the different stations caused by a particular truck sequence under consideration:

We define:

o_{it} ... overlap in station i in cycle t (lateness of work completion in station i)

Since o_{it} can be positive or negative it is convenient to define:

x_{it} ... positive overlaps in station i at the end of cycle t (work load left over at the end of the cycle)

$$x_{it} = \max\{0; o_{it}\} \qquad \text{for all } \; t = 1, ..., T \text{ and } \; i = 1, ..., I \quad (1)$$

y_{it} ... negative overlaps in station i at start of cycle t (work load done before beginning of the cycle)

$$y_{i,t+1} = \max\{0; -o_{it}\} \qquad \text{for all } \; t = 1, ..., T-1 \; \text{ and } \; i = 1, ..., I \quad (2)$$

Note that if o_{it} is negative, then it is possible to start earlier working on the truck of cycle $t+1$ which is why we have $y_{i,t+1}$ rather than y_{it} in (2).

At first sight one might think that the overlap is simply the excess work $o_{it} = p_{it} - w_i$. However, due to possible positive and/or negative overlaps in the previous period(s) the work in station i need not start at the beginning of the cycle. Thus,

$$o_{it} = \underbrace{x_{i,t-1} - y_{it}}_{\text{starting overlap}} + \underbrace{p_{it} - w_i}_{\text{excess work}} \quad \text{for all} \quad t = 1, ..., T \quad \text{and} \quad i = 1, ..., I. \qquad (3)$$

As the maximum overlap is restricted this model may produce infeasible solutions in some problem instances. The allocation of floater is necessary for reducing positive and negative overlaps.

3.1.2 Time model with floater allocation

If the positive overlap in some station is too high, then floaters have to be allocated. We denote:

$\bar{x}_i$ ... maximum (positive) overlap permitted in station i

f_{it} ... number of floaters allocated to station i in cycle t. It is assumed, that these floaters are available in station i from the beginning of the cycle to the end (without any overlaps or travel times).

In order to compute the number of floaters necessary, we now call o_{it} from (3) the theoretical overlap or overlap with "$no\,floaters$", o_{it}^{nf}, i.e.,

$$o_{it}^{nf} = x_{i,t-1} - y_{it} + p_{it} - w_i \quad \text{for all} \quad t = 1, ..., T \quad \text{and} \quad i = 1, ..., I. \qquad (4)$$

If o_{it}^{nf} is positive and exceeds $\bar{x}_i$, then floaters have to be allocated here to take care (at least) of the excess work load $o_{it}^{nf} - \bar{x}_i$:

$$f_{it} = \max\left\{0; \left\lceil o_{it}^{nf} - \bar{x}_i \right\rceil\right\} \quad \text{for all} \;\; t = 1, ..., T \;\; \text{and} \;\; i = 1, ..., I, \tag{5}$$

where only an integer number of floaters[1] can be allocated. Thus, usually f_{it} will be larger than $o_{it}^{nf} - \bar{x}_i$. Consequently this yields actual overlaps o_{it} which can be positive or negative, since the "excess work after allocation of floaters", $p_{it} - w_i - f_{it}$, can now be negative:

$$o_{it} = \underbrace{x_{i,t-1} - y_{it}}_{\text{starting overlap}} + \underbrace{p_{it} - w_i - f_{it}}_{\text{excess work}} \text{for all} \;\; t = 1, ..., T; i = 1, ..., I. \tag{6}$$

As in (1) and (2) we obtain the actual positive and negative overlaps x_{it} and y_{it} as

$$x_{it} = \max\{0; o_{it}\} \quad \text{for all} \;\; t = 1, ..., T \;\; \text{and} \;\; i = 1, ..., I, \tag{7}$$

$$y_{i,t+1} = \max\{0; -o_{it}\} \quad \text{for all} \;\; t = 1, ..., T \;\; \text{and} \;\; i = 1, ..., I, \tag{8}$$

or rather, if the negative overlap is restricted by an upper bound $\bar{y}_i$, as

$$y_{i,t+1} = \min\left\{\max\{0; -o_{it}\}, \bar{y}_i\right\} \quad \text{for all} \;\; t = 1, ..., T \;\; \text{and} \;\; i = 1, ..., I. \tag{9}$$

Now clearly the time model can be solved forward in time by simply using the recursion:

Forward recursion for the earliest starting (or end) times

1. Start with $x_{i,0} = 0$ and $y_{i,1} = 0$ for all $i = 1, ..., I$

[1] As usual, we use $\lceil X \rceil$ to denote the "smallest integer value greater than or equal to x".

2. For $t = 1$ to T:

   For $i = 1$ to I:

   compute o_{it}^{nf}, f_{it}, o_{it}, x_{it}, and $y_{i,t+1}$ using (4), (5), (6), (7), and (9).

The advantage of this simple procedure is that it is a very fast and efficient way to obtain the minimum number of floaters required. However, we note that one will always start as early as possible using as much negative overlap y_{it} as available, even if this is not necessary.

3.1.3 Reduction of negative overlap

After the time model has been solved using the above recursion, one knows the minimum number of floaters needed for each i and t. Given these, one can reconsider the actual starting times by reducing negative overlaps whenever they are not needed to avoid additional floaters. The procedure is the same as in project planning, where the forward procedure (4), (5), (6), (7), and (9) corresponds to the computation of the earliest starting and end times, while we now have to determine the latest end and starting times. This, of course, has to be done backward in time.

We use upper case letters O_{it}, X_{it}, and $Y_{i,t+1}$ in order to denote overlaps, positive and negative overlaps *after* the reduction of negative overlap:

Backward recursion for the latest starting (or end) times

1. Start with $O_{iT} = X_{iT} = x_{iT}$ as obtained from the forward recursion (for all $i = 1, ..., I$)

   i.e. unnecessary negative overlap in the (artificial) period $T+1$ is eliminated[2].

[2] Alternatively, one could start from the maximum permitted positive overlap $O_{iT} = \bar{x}_i$ in order to obtain the maximum buffer for each job in each station.

2. For $t = T$ to 1: (backwards in time)

   For $i = 1$ to I:

   $O_{i,t-1} = \min\{O_{it} - p_{it} + w_i + f_{it}, \bar{x}_i\}$

   with f_{it} as obtained from the forward recursion above.

The actual positive and negative overlaps X_{it} and Y_{it} are obtained as in (7) and (9). By using this approach, one obtains for each job i and each cycle t a *buffer* $O_{it} - o_{it}$ representing the slack in the starting or end time of the job in the station without violating any constraints or causing additional floaters.

It is also possible to choose starting (or end) times between the earliest and latest possibilities as follows:

Backward recursion for intermediate starting (or end) times

1. Start with any $\tilde{O}_{iT}$ between o_{iT} as obtained from the forward recursion and the maximum possible value $\bar{x}_i$. (for all $i = 1, ..., I$)

2. For $t = T$ to 1: (backwards in time)

   For $i = 1$ to I:

   Choose $\tilde{O}_{i,t-1}$ between the minimum value $o_{i,t-1}$ (obtained from the forward recursion) and the maximum value $\min\left\{\tilde{O}_{it} - p_{it} + w_i + f_{it}, \bar{x}_i\right\}$.

Clearly, the value of the objective function "total number of floater cycles" is the same in all variants o_{it}, $\tilde{O}_{it}$, and O_{it}. Since the solution o_{it} from the forward recursion is optimal, all these solutions are optimal. In this sense one should be careful w.r.t. the choice of the appropriate objective function. Only considering "total number of floater cycles" will yield many optimal solutions and also will ignore the fact that positive and negative overlaps should also be avoided.

This will be discussed in the next subsection.

3.1.4 LP-model minimizing the *W*age *C*osts of *F*loaters

Rather than fixing all the relevant variables by postulating (4), (5), (6), (7) and (9) we now only consider the constraints which have to be fulfilled by a feasible solution:

$$o_{it} \geq o_{i,t-1} + p_{it} - w_i - f_{it} \quad \text{for all } t = 1, ..., T \text{ and } i = 1, ..., I \tag{10}$$

$$o_{it} \geq -\bar{y}_i \quad \text{for all } t = 1, ..., T \text{ and } i = 1, ..., I \tag{11}$$

$$o_{it} \leq \bar{x}_i \quad \text{for all } t = 1, ..., T \text{ and } i = 1, ..., I \tag{12}$$

$$f_{it} \geq 0 \quad \text{and integer} \quad \text{for all } t = 1, ..., T \text{ and } i = 1, ..., I \tag{13}$$

where the (artificial) starting overlap is zero: $o_{i0} = 0$ for all $i = 1, ..., I$. Clearly, (10) is the counterpart of (6).

As usual, because of (11), it is more elegant to change variables from o_{it} ... actual overlap in station i and period t to $z_{it} = o_{it} + \bar{y}_i$ ... lateness compared to starting with maximum negative overlap in station i and period t.

Then, (10) to (13) can be written as

$$z_{it} \geq z_{i,t-1} + p_{it} - w_i - f_{it} \tag{14}$$

$$z_{it} \geq 0 \tag{15}$$

$$z_{it} \leq \bar{x}_i + \bar{y}_i \tag{16}$$

$$f_{it} \geq 0 \quad \text{and integer} \tag{17}$$

for all $t = 1, ..., T$ and $i = 1, ..., I$, where $z_{i0} = \bar{y}_i$ for all $i = 1, ..., I$.

Clearly, if one wants to minimize the number of "floater cycles", one obtains the objective function:

$$\sum_{t=1}^{T}\sum_{i=1}^{I} f_{it} \rightarrow \min \tag{18}$$

This is equivalent to saying that the Wage Costs of Floaters are to be minimized:

$$WCF = \omega_0 \sum_{t=1}^{T}\sum_{i=1}^{I} f_{it} \rightarrow \min \tag{19}$$

where ω_0 is the "wagerate" of a floater per cycle (assuming that in cycles where a floater is not used he can do productive work elsewhere). In this sense ω_0 can also be seen as opportunity cost of allocating one floater one cycle.

Clearly, the simple forward recursion (4), (5), (6), (7), and (9) from the previous subsection (as well as the backward recursions) yields an optimal solution of the LP (19) s.t. (10) to (13) or (14) to (17).

As noted above, this solution has the (often undesired) property that negative overlaps are used even when not necessary. However, if a worker has to work outside his own station this causes some problems with availability of tools, work satisfaction etc.

3.1.5 LP-models with various objectives

The LP (19) s.t. (10) to (13) or (14) to (17). in general has multiple optimal solutions in the sense that the buffer between earliest and latest start and end times for each activity can be used. In order to single out the most desirable among all solutions, which are optimal w.r.t. (19) one can use a lexicographic approach by defining the second objective "minimize the Weighted Sum of Actual Overlaps".

$$WSAO = \omega_1 \sum_{t=1}^{T}\sum_{i=1}^{I} x_{it} + \omega_2 \sum_{t=1}^{T}\sum_{i=1}^{I} y_{it} \rightarrow \min \tag{20}$$

where ω_1 and ω_2 measure the relative harm caused by positive and negative overlaps, respectively. Since in (20) the positive and negative parts x_{it} and $y_{i,t+1}$ of the actual overlap o_{it} are explicitly used, they have to be computed in the model. This is done by simply adding the constraints

$$x_{it} \geq o_{it} \tag{21}$$

$$y_{i,t+1} \geq -o_{it} \tag{22}$$

$$x_{it} \geq 0, \quad y_{i,t+1} \geq 0 \tag{23}$$

for all $t = 1, ..., T$ and $i = 1, ..., I$ to the LP (10) to (13) or

$$x_{it} \geq z_{it} - \bar{y}_i \tag{24}$$

$$y_{i,t+1} \geq \bar{y}_i - z_{it} \tag{25}$$

$$x_{it} \geq 0, \quad y_{i,t+1} \geq 0 \tag{26}$$

for all $t = 1, ..., T$ and $i = 1, ..., I$ to the LP (14) to (17).

Rather than using a lexicographic approach, one can also consider the Total Cost as the sum of WCF and $WSAO$:

$$TC = \omega_0 \sum_{t=1}^{T} \sum_{i=1}^{I} f_{it} + \omega_1 \sum_{t=1}^{T} \sum_{i=1}^{I} x_{it} + \omega_2 \sum_{t=1}^{T} \sum_{i=1}^{I} y_{it} \rightarrow \min \tag{27}$$

where it may be difficult to estimate the values of ω_1 and ω_2 as the cost caused by one unit of overlap in one station.

Remark: We could drop the assumption that floaters are used in a productive way (e.g. in quality control or in the manual assembly of exotic parts after the line) in cycles when they are not used as floaters. Rather, we assume that the cost are caused not by using floaters in particular cycles but by having them. In this sense

we have a capacity problem and a floater causes the same cost if he is used in just one cycle or if he is used all the time. In this situation the term $\sum_{t=1}^{T}\sum_{i=1}^{I} f_{it}$ in the above objectives has to be replaced by

$$T \cdot F \tag{28}$$

where the variable F denotes the maximum number of floaters allocated in any cycle. This F is obtained in the above LPs by adding the constraints

$$\sum_{i=1}^{I} f_{it} \leq F \qquad t = 1, ..., T. \tag{29}$$

$$F \geq 0 \tag{30}$$

3.1.6 Model variant with conflicting overlaps

In case of conflicting overlaps, work in a certain station i in a cycle t can only be started when

- the work on the previous item (truck in cycle $t-1$) in station i has been completed (as before) *and*
- the work on the current item (truck in cycle t) has been completed in the previous station $i-1$ (there being the truck in cycle $t-1$); see also Figure 2.

Thus, in our simple forward recursion, we have to replace (4) and (6) by

$$o_{it}^{nf} = \max\{x_{i,t-1} - y_{it}, x_{i-1,t-1} - y_{i-1,t}\} + p_{it} - w_i, \tag{31}$$

and

$$o_{it} = \max\{x_{i,t-1} - y_{it}, x_{i-1,t-1} - y_{i-1,t}\} + p_{it} - w_i - f_{it}. \tag{32}$$

In the backward recursion, we have to replace $O_{i,t-1} = \min\{O_{it} - p_{it} + w_i + f_{it}, \bar{x}_i\}$ by

$$O_{i,t-1} = \min\{O_{it} - p_{it} + w_i + f_{it}, O_{i+1,t} - p_{i+1,t} + w_{i+1} + f_{i+1,t}, \bar{x}_i\}$$

which can be seen best by locking at (10) and (33) below. In the LP-models above, one has to add the constraints

$$o_{it} \geq o_{i-1,t-1} + p_{it} - w_i - f_{it} \tag{33}$$

and

$$z_{it} - \bar{y}_i \geq z_{i-1,t-1} - \bar{y}_{i-1} + p_{it} - w_i - f_{it} \tag{34}$$

for all $t = 1, ..., T$ and $i = 1, ..., I$ to (10) to (13) and (14) to (17), respectively.

3.2 Floater Allocation

After we have determined the floater time requirements f_{it}, we can find a schedule for each of the floaters needed. The minimum number of floaters required, F, can be calculatd as follows:

$$F = \max_t \sum_{i=1}^{I} f_{it} \tag{35}$$

The model for floater allocation is formulated to minimize the overall tranfer times for the floaters when they are moving from one station to any other. We define a dummy station $i = 0$ where the floaters have to perform when they are not needed at the assembly line. In our model the binary variable $r^t_{f,i}$ indicates that floater f is allocated to station i during cycle t. Moreover, in order to avoid nonlinearities

we introduce R_{fi}^{jt} which should be 1 if floater f moves from station i to station j between cycle $t-1$ and t. Let d_{ij} denote the distance between stations i and j. d_{0i} should be chosen reasonable high in order to generate compact schedules for the floaters. The mathematical optimization problem can be formulated as:

$$\sum_{t=1}^{T}\sum_{f=1}^{F}\sum_{j=0}^{I}\sum_{i=0}^{I} d_{i,j} \cdot R_{fi}^{jt} \rightarrow \min \tag{36}$$

$$R_{fi}^{jt} \geq r_{fj}^{t} + r_{fi}^{t-1} - 1 \qquad \text{for all } f = 1,\ldots,F; \quad i,j = 0\ldots I; \quad t = 1,\ldots,T \tag{37}$$

$$\sum_{f=1}^{F} r_{fi}^{t} = f_{it} \qquad \text{for all } i = 1,\ldots,I \text{ and } t = 1,\ldots,T \tag{38}$$

$$\sum_{i=0}^{I} r_{fi}^{t} = 1 \qquad \text{for all } f = 1,\ldots,F \text{ and } t = 1,\ldots,T \tag{39}$$

$$R_{fi}^{jt} \geq 0 \tag{40}$$

$$r_{fi}^{t} \in \{0,1\} \tag{41}$$

where $r_{f0}^{0} = 1$ and $r_{fi}^{0} = 0$ for $i \neq 0$.

Since the above MIP is of very high dimension, we rather use a heuristic approach by applying the shortest distance rule, where the floater with the highest utilization is scheduled first.

3.3 Truck Sequencing

The floater requirements are clearly sequence dependent. It would be desirable to have a sequence available so that in each station and cycle a truck with much excess

workload $p_{it} - w_i$ is preceeded and/or followed by a truck with moderate workload so that positive and negative overlaps can be used rather than allocating floaters. Indeed, the motivating local truck manufacturer is applying a commercial software package which aims at constructing a sequence based on the mean processing times of three subsequent items.

Although we can easily identify a good production sequence at a particular station, considering all stations of a production line makes the problem much more demanding. For the overall system we have to find a sequence which gives the minimum floater time requirements. We can evaluate this particular sequence according to the objective function (19) and (27), respectively, by using the time model recursions. These calculations are very fast so that we are able to evaluate a large number of sequences during the heuristic. In our case the resulting time requirements (distance) for a particular item (truck) are depending on the order sequence fixed so far. Thus, for developing a solution approach for production sequencing we can follow the ideas of well known TSP-heuristics with appropriate modifications due to the sequence dependence of the transition costs of the 'cities' of this 'TSP'.

For obtaining good starting solutions, any well known opening heuristics can be used such as "nearest neighbor" or "successive insertion" with obvious modifications (e.g. the cost effect of an insertion cannot simply be obtained by adding and subtracting transition costs, but a part of the time model of section 3.1. has to be recomputed).

Also, when improving a solution by applying some hill climbing procedure (e.g. 2-opt) or simulated annealing, genetic algorithms, tabu search, etc. one has to recompute the part of the time model which follows the first change in the original sequence, in order to evaluate the new solution. Since this is however straightforward, we refrain from explaining these details here.

3.4 Worker Allocation

The overall workforce consists of workers and floaters. Workers are restricted to some adjacent stations when performing their work. On the other hand, floaters can work at any station. Further, workers will stay at their station during the whole shift once they are allocated as depicted in Figure 4. With this respect the worker allocation is sequence independent. The goal of worker allocation is to minimize the maximum overlap at a particular station.

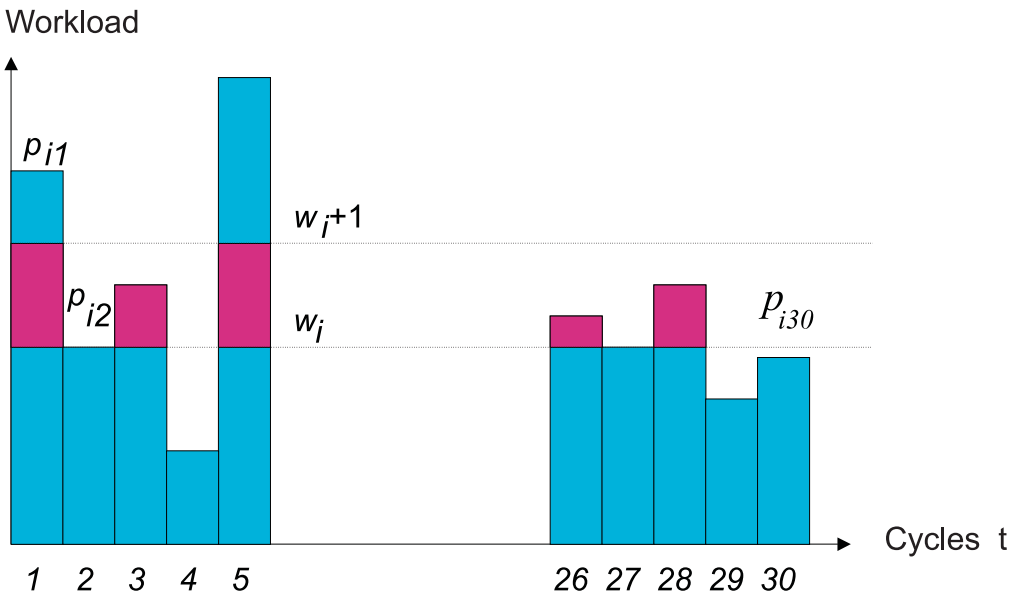

Figure 4: Worker allocation proplem

In order to allocate the appropriate number of workers to each station we can safely start by allocating the minimal number of workers

$$w_i = \left\lfloor \min_{t=1...T} p_{it} \right\rfloor \tag{42}$$

to each station $i = 1, ..., I$. These workers will never be idle if allocated[3].

[3]As usual, the notation $\lfloor X \rfloor$ means "largest integer value less than or equal to X".

Then, one can allocate the remaining workers according to some reasonable priority rule.

1. For $i = 1, ..., I$ allocate w_i from (42)
2. As long as workers are available for allocation:
   (re)compute the priority numbers b_i according to one of the rules below
   allocate the next worker to station i with the highest priority number b_i and increase this w_i by 1.

The following rules seem reasonable:

Rule 1: *T*otal *M*arginal *W*ork *L*oad ($TMWL$)

$$b_i = \sum_t \max\{0; \min\{p_{it}; w_i + 1\} - w_i\}$$

for $i = 1, ..., I$, which can also be written as

$$b_i = \sum_t \max\{0; \min\{p_{it} - w_i; 1\}\}$$

This quantity is the sum of the dark shaded areas in Figure 4. Note that the marginal work load (additional work load done because of the additional worker) can only be positive, if the currently allocated w_i workers cannot cope with work load p_{it} in cycle t, i.e., $p_{it} > w_i$. On the other hand, it can never exceed 1 or fall below 0.

Rule 2: *N*umber of *C*ycles *A*ctive (NCA)

$$b_i = \|\{t \mid 1 \leq t \leq T,\ p_{it} > w_i\}\|$$

for $i = 1, ..., I$, where $\|\cdot\|$ denotes the cardinality of a set. According to this rule, workers are allocated in stations where they will not be completely idle in as many cycles as possible.

Rule 3: *M*aximum *A*mount of *E*xcess *W*ork ($MAEW$)

$$b_i = \max_{t=1...T} \{p_{it} - w_i\}$$

for $i = 1, ..., I$. This rule may be useful if otherwise the number of floaters available is not sufficient there.

As usual when applying priority rules, it is also possible to combine the various priority measures by taking (weighted) sums or by applying different rules in different phases of the algorithm. In Section 4 we will report on numerical investigations of an industry problem where the application of the rules $TMWL$ and $MAEW$ in different phases of the algorithm worked best.

The various WA-rules are evaluated by the following objective function (43) which considers the number of workers and the floater time allocation. This function is an extension of the former formulated function in (19). By using this evaluation function we can easily find the cost minimal allocation of workers scheduled for the line and the required floater time.

$$WCWF = \sum_{i=1}^{I} w_i + \omega_0 \sum_{t=1}^{T} \sum_{i=1}^{I} f_{it} \rightarrow \min \tag{43}$$

Further, we can complete our models from Section 3.1.4 by inserting the worker pool restriction, for a given pool size of W.

$$\sum_{i=1}^{I} w_i \leq W \tag{44}$$

4 Computational results

The data for this numerical study were provided by a local truck manufacturer. The considered assembly line consists of 22 different stations, including six painting stations which we can ommit in our analysis. The processing time variabilites from a sample of nine trucks are depicted in Figure 5. We can state that we are faced with both a high processing time variability betweeen the stations and within a particular station (see e.g. station 7). In our numerical study we want to figure out the quality of the different WA-rules and the solution quality of our overall approach.

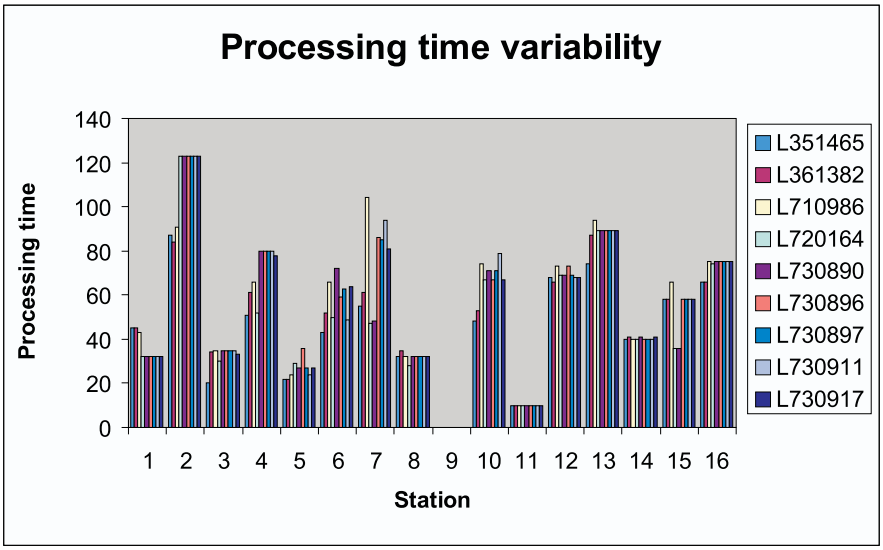

Figure 5: Processing time variabilities

Furthermore, our results are compared to the solutions generated by the shop floor manager. The real data set used contains the production mix of 20 subsequent days. Daily, 44 items are assembled on the line. According to the real environment we have to consider open and closed stations. Worker's overlap is limited to one station.

In a number of pre-test experiments for TS we investigated several local search procedures like Tabu search, Simulated Annealing and Hill Climbing. As a result, we found that with respect to this step of the overall solution approach, there is only very small room for improvement. In our analysis we therefore decided to choose a hill climbing procedure because of its speed without loosing solution quality significantly.

4.1 Worker Allocation Rules

In Section 3.4 we have presented three different rules for worker allocation. These are: TMWL, NCA and MAEW respectively. Moreover, we also applied a combined procedure. Starting with the TMWL rule we switch to the MAEW rule if the marginal work load of the next worker to allocate drops to 33.3%, 50% and 75% respectively. We use the evaluation function (43) in order to figure out the most appropriate WA-rule for different sizes of the worker pool. ω_0 is set to 1.05. The results for the average costs per day in Figure 6 indicate a similar behaviour of all our rules except rule MAEW. We can also observe that at the near optimal region the relevant rules generate similar good results.

4.2 Floater Time Allocation

In our second series of study we compare our suggested approach with the solution of the shop floor manager. Again, the same data set as above was used. In addition, the management could provide the actual daily worker pool size ($W = 33$) and the launching sequence of the products. We calculated the floater time allocation given these already fixed parameters. In our approach we used the same worker pool size,

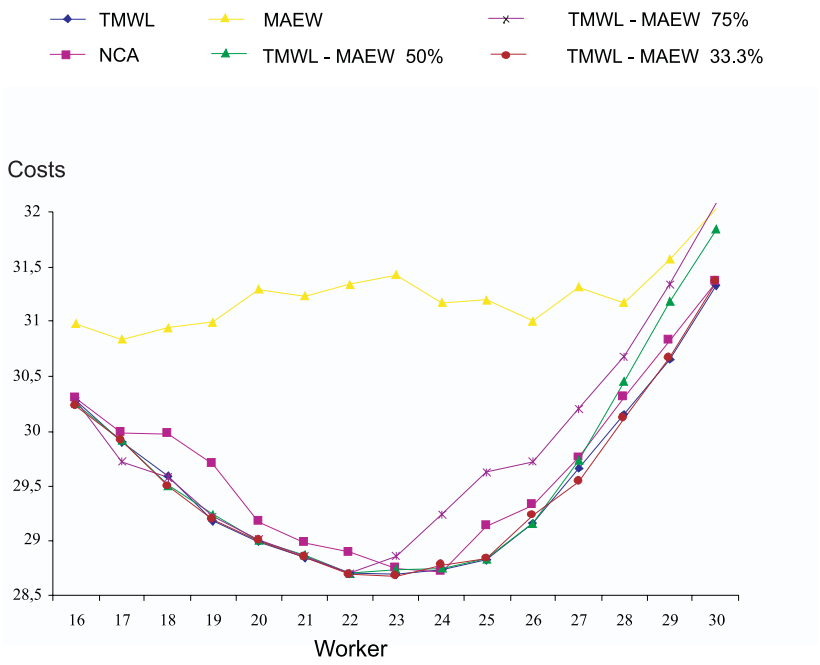

Figure 6: Worker Allocation rules

but we modifying the TS and the FTA according to the models in Section 3.3 and 3.1.

The results indicate an average reduction of floater costs of 85.2% compared to the actual situation. The number of floater can also be reduced about 50%. Knowing the results of Section 4.1, where we found that the worker pool size should be about 23, the large differences between the shop floor solution and our proposed approach in Figure 7 are not surprising. Additionally, due to labor laws extensive worker and floater movements are not allowed and shop floor management cannot reduce negative overlaps systematically. The results of our theoretical models can be seen as a bottom line for improving shop floor performance.

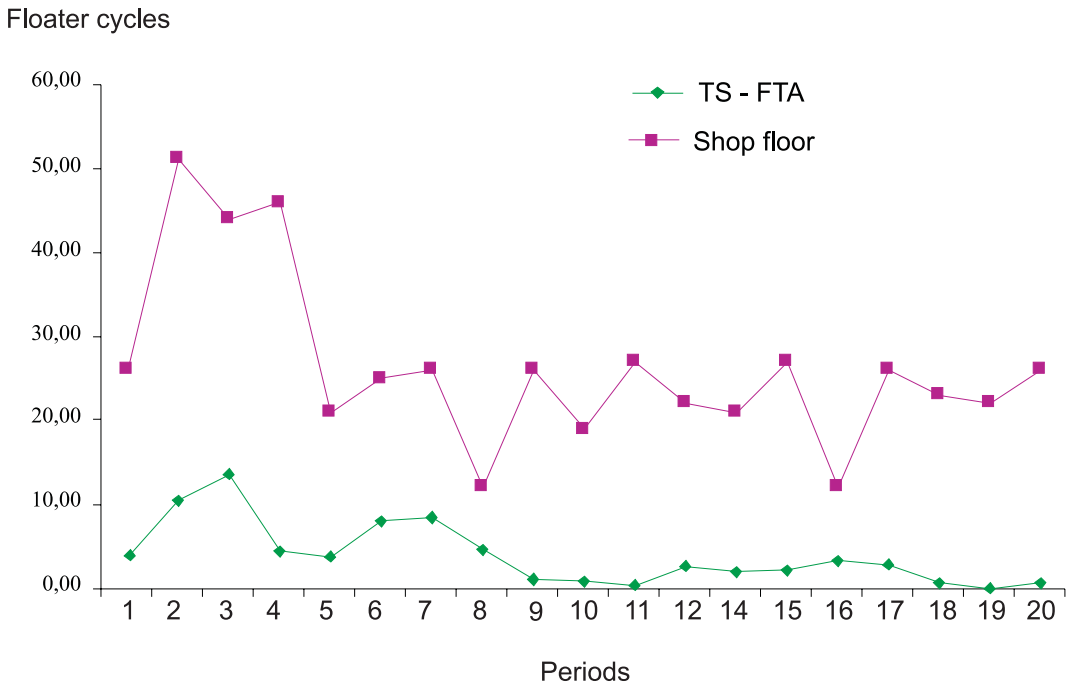

Figure 7: Floater time allocation

5 Conclusions

In this paper we have developed a comprehensive approach for worker and floater time allocation in a mixed-model assembly line. In detail we have outlined various time models and have proposed some solution procedures both for each respective subproblem and the overall problem. The obtained results show that advanced planning procedures may guide shop floor managers to improve their performance. Future works should address the interaction between weekly production planning issues and the daily sequencing of products in order to reduce the processing time variabilities.

References

[1] J. F. Bard, E. Dar-El, and A. Shtub. An analytic framework for sequencing mixed model assembly lines. *Int. J. Prod. Res.*, 30(1):35–48, 1992.

[2] J. Bartholdi III and D. Eisenstein. A production line that balances itself. *Operations Research*, 44(1):21–34, 1996.

[3] M. Decker. Capacity smoothing and sequencing for mixed-model lines. *Int. Journal of Production Economics*, 30-31:31–42, 1993.

[4] C. Lee and G. L. Vairaktarakis. Workforce planning in mixed model assembly systems. *Operations Research*, 45(4):553–567, 1997.

[5] G. Vairaktarakis and J. K. Winch. Worker cross-training in paced assembly lines. *Manufacturing & Service Operations Management*, 1(2):112–131, 1999.