A Heuristic Multi-Start Decomposition Approach for Optimal Design of Serial Machining Lines

O. Guschinskaya¹, A. Dolgui¹*, N. Guschinsky², and G. Levin²

¹Ecole des Mines de Saint Etienne, 158, cours Fauriel 42023 Saint Etienne Cedex 2 France
²United Institute of Informatics Problems, Surganov Str, 6, 220012 Minsk  Belarus

Abstract: We study an optimal design problem for serial machining lines. Such lines consist of a sequence of stations. At every station, the operations to manufacture a product are grouped into blocks. The operations within each block are performed simultaneously by the same spindle head and the blocks of the same station are executed sequentially. The inclusion and exclusion constraints for combining operations into blocks and stations as well as the precedence constraints on the set of operations are given. The problem is to group the operations into blocks and stations minimizing the total line cost. A feasible solution must respect the given cycle time and all given constraints. In this paper, a heuristic multi-start decomposition approach is proposed. It utilizes a decomposition of the initial problem into several sub-problems on the basis of a heuristic solution. Then each obtained sub-problem is solved by an exact algorithm. This procedure is repeated many times, each time it starts with a new heuristic solution. Computational tests show that the proposed approach outperforms simple heuristic algorithms for large-scale problems.

Keywords: Machining Line; Logical Layout Design; Combinatorial Optimization; Graph Approach; Mixed Integer Programming; Decomposition; Heuristics.

* Corresponding author: dolgui@emse.fr
1. Introduction

Serial machining lines form the heart of many manufacturing systems (Hitomi, 1996; Dashchenko, 2003) and are used for the production of a single product or a family of similar products. We study an optimal design problem for such lines. The design purpose is to determine the logical line layout, i.e. define the number of required production facilities and equipment. Finding a good (and if possible the optimal) design solution is worthwhile because of the large investment cost (from 5 million to 50 million euros for a line) and long exploitation period (approximately 7 years without reconfiguration and in use about 30 years).

We consider the type of lines where a part is processed successively on several (1, …, m) linearly ordered stations without buffers between. The part is moved from station to station by an automatic material handling device. The time span between two movements is referred to as cycle time $T_0$.

To respect the cycle time objective, the total processing time of all operations executed at each station $t^r$ must be less than the cycle time value (i.e. $t^r$ must be inferior or equal to $T_0$). Loading and unloading of the part at all stations must be executed within the same cycle time, the loading and unloading time $\tau_s$ can be considered the same for all stations.

The manufacturing process is defined by a set of machining operations $N$ required for production of a part. Each machining operation deals with removal of material (drilling, milling, boring,…). In order to carry out an operation, a cutting tool is needed.

The machining is done by tools fixed at the spindle heads. Each machine is equipped with several spindle heads. The activation of spindle heads at the same machine is sequential, due to the necessity to change the active spindle head or to move (reposition) the part. The reposition time can be assumed the same for all spindle heads, and equal to $\tau^b$. The number of spindle heads at each station and the order of their activation must be determined during the line design process. Each spindle head incurs an
investment cost and requires a certain amount of working space, therefore the total number of spindle heads which can be assigned to the same station is limited by number \( n_0 \).

Each operation \( j \in N \) can be characterized by two parameters: the required working stroke length \( \lambda_j \) and the recommended feed per minute \( s_j \). The working stroke length includes the required depth of cut and the distance between the tool and the part surface. The feed is measured by how much the tool advances for each rotation of the tool.

Each spindle head may have multiple tools to perform all the operations assigned to it simultaneously. If there is only one tool fixed in the spindle head which executes only one operation, e.g. operation \( j \), then the feed per minute \( S(N) \) of the spindle head is equal to \( s_j \) and its working stroke length \( L(N) \) is equal to \( \lambda_j \). If there are several tools performing a set of operations \( N \subseteq N \) then the feed per minute of the spindle head is equal to

\[
S(N) = \min \{ s_j | j \in N \}
\]

and its working stroke length is equal to

\[
L(N) = \max \{ \lambda_j | j \in N \}.
\]

A set of operations grouped together to be executed simultaneously by the same spindle head is referred to as \textit{block of operations} (or \textit{block}).

There may exist incompatibilities among groups of operations, which prohibit machining a given group of operations by the same spindle head or at the same station (e.g., because tool location or manufacturing incompatibility). We referred to these types of constraints as \textit{block or station exclusion constraints}, respectively.

There also exist groups of operations that must be executed all without repositioning of part, to avoid a loss of the required precision. These operations must be assigned to the same station in any feasible solution. We refer to this type of constraint as \textit{station inclusion constraints}. (Note: the operations to be
assigned to the same spindle head, if exist, can be replaced by a macro-operation, see Dolgui et al., 2000).

Finally, there are precedence constraints between operations.

At the preliminary design stage, we search to determine an assignment of operations to a sequence of stations \((k=1, 2, \ldots, m)\) and repartition of operations assigned to the same station \(k\) to \(n_k\) blocks. Let \(N_k\) be the work content (i.e. a set of operations) for station \(k\) \((k=1, 2, \ldots, m)\) and \(N_{kl}\) be the set of operations grouped into common block \(l\) \((l=1, 2, \ldots, n_k)\) of station \(k\).

Let \(P=\{\{N_1, \ldots, N_{n_1}\}, \ldots, \{N_m, \ldots, N_{n_m}\}\}\) be a design decision. To be feasible, it must respect the given cycle time and satisfy all above constraints.

Two weights \(C_1\) and \(C_2\) are assumed to be given. They can be interpreted as the relative cost of a station and a spindle head, respectively. The line designer can change these coefficients to take into account the relative importance of station or spindle head in a specific project.

The goal is to find an optimal solution \(P\) which minimizes the following criterion:

\[
Q(P) = C_1 m + C_2 \sum_{k=1}^{m} n_k ,
\]

where

- \(m\) is the number of stations,
- \(n_k\) is the number of blocks (spindle heads) assigned to station \(k\).

We refer this problem to as the Transfer Line Balancing Problem (TLBP) by analogy with Assembly Line Balancing Problem (ALBP).

The remainder of the paper is organized as follows. In Section 2 we compare this problem with those in literature. Section 3 introduces a mathematical model. Section 4 presents our optimization approach. Section 5 is dedicated to experimental results. Conclusions are presented in Section 6.
2. Literature review and previous work

TLBP is related to the simple assembly line balancing problem (SALBP) (Baybars, 1986; Scholl and Klein, 1998). The SALBP consists in assigning a set of operations to identical consecutive stations minimizing the number of stations required, subject to precedence (between operations) and cycle time constraints. The principal approaches used in literature for SALB are: Lagrange relaxation techniques (Aghezzaf and Artiba, 1995), Branch and Bound algorithms, see for example (Assche and Herroelen, 1998; Ugurdag et al., 1997; Scholl and Klein, 1998), and heuristics and meta-heuristics (Arcus, 1966; Helgenson and Birnie, 1961; Rekiek et al., 2001). A state-of-the-art is presented in (Baybars, 1986; Becker and Scholl, 2006; Erel and Sarin, 1998; Ghosh and Gagnon, 1989; Rekiek et al., 2002).

When the line balancing problem is solved jointly with equipment selection, this is often called Simple Assembly Line Design Problem (SALDP) (Baybars, 1986; Becker and Scholl, 2006) or Single-Product Assembly System Design Problem (SPASDP) (Gadidov and Wilhelm, 2000).

In (Gadidov and Wilhelm, 2000) the total cost is estimated as the sum of the fixed cost of stations and machines and the variable cost of assembly operations over the planning horizon. There is a set of available machines for each operation. The processing time of each operation on each machine as well as the corresponding cost are given. The operations must be assigned to stations; each station can be equipped with only one machine. The goal is to minimize the total cost.

The authors also considered two extensions of this SPASDP. The first with parallel, identical machines at each station, the second considers positional constraints caused by the impossibility of assigning to the same station front and back side operations. For these three SPASDP, the authors proposed a branch-and-cut approach, which is complemented by some preprocessing methods and a heuristic to obtain an upper bound.
Bukchin and Tzur (2000) studied a SPASDP for a flexible assembly line. This problem consists of selecting equipment and assigning operations to each station. The objective is to minimize total equipment cost. Due to the flexibility of the available pieces of equipment, there are several equipment alternatives for each operation, and often a particular piece of equipment is efficient for some tasks, but not for others. So there is a given set of equipment types, each type is associated with a specific cost. The equipment cost is assumed to include the purchasing and operational cost. The duration of an operation depends on the equipment selected. An operation can be performed on any station, provided that the equipment selected for this station is appropriate and that precedence relations are satisfied. The total station time should not exceed the predetermined cycle time. The authors developed a branch and bound algorithm, which solves problems of moderate size. The algorithm's efficiency was enhanced due to appropriate lower bounds, as well as the use of some dominance rules to reduce the size of the branch and bound tree. The authors also suggested a branch-and-bound based heuristic procedure for large problems.

Bukchin and Rubinovitz (2003) focus on station paralleling. Two problems with objective of minimizing the number of stations and the total cost were discussed. The authors showed that the problem of assembly system design with parallel stations can be treated as a special case of the problem with stations in series (Bukchin and Tzur, 2000).

More details on other similar approaches in the literature can be found in the recent survey on generalized assembly line balancing problems (Becker and Scholl, 2006).

Dolgui et al. (2006) demonstrated that TLBP is more complex than SALBP and SALDP because the operations must be grouped into blocks of parallel operations. The known optimization methods cannot be used directly for this problem due to the following reasons:

- TLBP deals with “low-level” logical layout where it is necessary not only to define a required number of stations but also to design each station by grouping operations into a number of blocks. In
contrast, SALDP considers the “high-level” logical layout design with equipment selection (one per station) from a set of alternatives.

- Each block requires a piece of equipment (a multi-spindle head), which incurs a purchase cost. Therefore, it is necessary to minimize both the number of stations and the number of blocks. To do it all possible operations assignments to blocks and stations must be considered, otherwise the optimality of a solution cannot be guaranteed.

- The set of alternative blocks is not known in advance and the parameters of a block depend on the set of operations assigned to it.

- Usually, for the machining operations non-strict precedence constraints are used. Here, cases when machining operations, in spite of being linked by a precedence constraint, can be executed by the same compound tool simultaneously (for assembly systems this is forbidden).

In our previous works, several exact methods were suggested for this novel problem (TLBP):

a) An efficient method was suggested in (Dolgui et al., 2000), which transforms the initial problem into a constrained shortest path problem (see Subsection 4.4). Some dominance rules were developed to reduce the size of the obtained graph. More recent results on this method are given in (Dolgui et al., 2007).

b) A mixed integer programming (MIP) approach was developed in (Dolgui et al., 2000) dealing with the special cases of TLBP where all the operation times are given in advance (before grouping into blocks) and the block processing time is equal to the time of its longest operation. An improved and advanced MIP approach was presented in (Finel, 2004; Dolgui et al., 2006). More realistic (and more complex) assumptions and constraints, considering operation times given by the required working stroke and the recommended feed per minute, were introduced.
Since TLBP is NP-hard, the exact algorithms are only applicable for small and medium-sized problems (less than 60 operations). For large-scale problems, heuristic approaches were proposed in the following papers:

a) Two heuristic algorithms RAP and FSIC were suggested in (Finel, 2004; Dolgui et al., 2005). These heuristics are based on the COMSOAL method (Arcus, 1966), i.e. several operation assignments (in the increasing order of block and station indices) are randomly generated, and the best assignment is kept. The two heuristic algorithms RAP and FSIC (see Subsection 4.2) differ mainly in operation selection for assignment to blocks and stations.

b) An original technique of deterministic decomposition for MIP model was suggested in (Finel, 2004; Dolgui et al., 2006). In this paper, we present a novel approach, which is based on a stochastic multi-start decomposition of the problem into several sub-problems, which are further solved exactly by the shortest path algorithm. The decomposition is made on the basis of a feasible solution of the initial problem obtained by random search heuristic FSIC. This method does not guarantee an optimal solution but does improve the performance of the heuristic used individually.

3. Mathematical model

The aforementioned constraints of this problem are modeled as follows:

- Precedence constraints are represented by digraph $G = (N, D)$. There exist an arc $(i, j) \in D$ iff operation $j$ cannot be executed before operation $i$. Such precedence constraints are called non strict because if operation $i$ is a predecessor of operation $j$ then either operation $i$ can be executed before $j$ or $i$ and $j$ can be executed simultaneously.
Inclusion constraints are modeled by $ES$, a family of subsets from $N$, such that all operations of the same subset $e \in ES$ must be assigned to the same station. No two sets from $ES$ can include the same operation; otherwise such sets should be aggregated;

• Station exclusion constraints are modeled by $\overline{ES}$, a family of subsets from $N$, such that each subset $e \in \overline{ES}$ cannot be assigned to the same station as a whole;

• Block exclusion constraints are modeled by $EB$, a family of subsets from $N$, such that the same subset $e \in EB$ cannot belong to the same block as a whole.

Note 1: for the exclusion constraints any proper subset of a subset $e \in \overline{ES}$ (resp. $e \in EB$) can be assigned to a station (resp. block).

Note 2: in order to obtain the strict precedence constraint between $i$ and $j$ ($i$ strictly before $j$), it is necessary to include arc $(i, j)$ to set $D$ and the pair $\{i, j\}$ in $EB$.

The block processing time $t^b(N_{kl})$ is determined as follows: $t^b(N_{kl}) = \frac{L(N_{kl})}{S(N_{kl})} + \tau$. The spindle heads of the same station are activated sequentially, therefore the station processing time $t^s(N_k)$ is equal to the sum of its block processing times: $t^s(N_k) = \sum_{l=1}^{n_k} t^b(N_{kl}) + \tau$.

In order to reduce the number of feasible solutions we introduced a parameter $m_0$ which represents the maximal authorized number of stations. It can be obtained by analyzing the available plant area or the maximum authorized line investment cost, or calculated as an upper bound on $m$ taking into account all the constraints.

Thus, the considered transfer line balancing problem can be formulated as follows:

$$\text{Min } Q(P) = C_1m + C_2 \sum_{k=1}^{m} n_k,$$

subject to
The objective function (1) estimates the line cost. Expression (2) is the cycle time constraint. Constraints (3)-(4) reflect the fact that each operation from N must be included in a block once and only once. Constraints (5) define the precedence relations on the set N. Constraints (6) determine the necessity of assigning certain operations to the same station. Constraints (7)-(8) deal with the impossibility of grouping certain operations into the same block or executing certain operations at the same station, respectively. Constraints (9) and (10) limit the number of stations and blocks, respectively.

4. Optimization approach

4.1. General scheme of approach

An iteration of the suggested heuristic multi-start decomposition approach consists of four steps: 1) resolution of the initial problem by a heuristic; 2) decomposition of a heuristic solution into several sub-
problems; 3) solving the sub-problems by an exact method; 4) composition of a final solution via aggregation of optimal solutions of the sub-problems.

At the first step, the initial problem is solved by FSIC heuristic (see Subsection 4.2) to obtain a feasible solution $P_{\text{heur}}$, i.e. an assignment of all operations to $m_{\text{heur}}$ stations. This solution is a sequence of the sets of operations assigned to stations $k = 1, 2, \ldots, m_{\text{heur}}$. At the second step, this sequence is decomposed into several subsequences in such a way that:

- Each subsequence $r$ includes a number of consecutive stations;
- There are no intersections between subsequences;
- The union of all subsequences is the initial sequence.

At the third step, each subsequence $r$ of $k'$ stations is used to generate a new sub-problem $SP_r$ which consists in assigning the set $N^r$ of operations belonging to subsequence $r$. Each sub-problem $SP_r$ is solved exactly and independently of other sub-problems by the shortest path approach (see Subsection 4.4).

The scheme of this approach is shown in Fig. 1.

[Insert Figure 1 about here]

In this paper, the suggested approach is multi-start. It means that when an iteration is finished (a final solution is formed on the basis of partial solutions), a new iteration is started (searching for a new feasible solution) and so on until the end of available calculation time. This requires a new feasible heuristic solution each time. It can be reached by two ways: either using a new deterministic heuristic at each iteration or using a stochastic heuristic capable to give a different solution at each start. We choose the second variant because we have already multi-start heuristics developed for the considered problem.

In our realization, we use FSIC heuristic to obtain feasible solutions and the shortest path approach for exact solving the sub-problems but the developed approach can be also implemented with other algorithms. The quality of the obtained results depends on the quality of algorithms used. To ensure that
for different starts of FSIC heuristic the solutions are different at the end of each iteration, we change
the value of parameter seed which is initial value for generation of pseudo-random numbers.

Let us introduce the following notations:

\( P_{\text{min}} \) the best solution,

\( C_{\text{min}} \) the cost of the best solution,

\( TR_{\text{tot}} \) the current number of iterations,

\( TR_{\text{nimp}} \) the number of iterations without improving \( C_{\text{min}} \),

\( TR_{\text{tot.aut}} \) and \( TR_{\text{nimp.aut}} \) the maximum value of \( TR_{\text{tot}} \) and \( TR_{\text{nimp}} \) allowed,

\( T_{\text{cur}} \) the current solution time,

\( T_{\text{res}} \) the available solution time.

The general scheme of this heuristic multi-start decomposition approach is presented by Algorithm
1.

**Algorithm 1**

Set \( C_{\text{min}} = \infty \), \( TR_{\text{tot}} = 0 \), \( TR_{\text{nimp}} = 0 \), seed = 0.

while \( T_{\text{cur}} < T_{\text{res}} \) and \( TR_{\text{nimp}} < TR_{\text{nimp.aut}} \) and \( TR_{\text{tot}} < TR_{\text{tot.aut}} \)

Step 1. A current heuristic solution \( P_{\text{heur}} \) consisting of \( m_{\text{heur}} \) stations with cost \( C_{\text{heur}} \) is found by
FSIC heuristic.

If \( C_{\text{heur}} < C_{\text{min}} \) then \( C_{\text{min}} = C_{\text{heur}}, P_{\text{min}} = P_{\text{heur}}. \)

Set \( k = 1 \), \( r = 1 \).

while \( k < m_{\text{heur}} \)

Step 2. Choose \( k' \).

Generate a new sub-problem \( SP_r \) on the basis of \( N_r \).

Set \( k = k + k', r = r + 1 \).

Step 3. A sub-problem \( SP_r \) is solved by the shortest path method.

end while

Step 4. The solution of the initial problem \( P_{\text{hybr}} \) with cost \( C_{\text{hybr}} \) is formed on the basis of the
solutions of \( r \) solved sub-problems.
Set $TR_{tot} = TR_{tot} + 1$.

If $C_{hybr} < C_{min}$ then $C_{min} = C_{hybr}$, $P_{min} = P_{hybr}$, $TR_{nimp} = 0$, else $TR_{nimp} = TR_{nimp} + 1$.

Change seed.

end while

The output of Algorithm 1 is solution $P_{min}$.

Steps 1-4 of Algorithm 1 are detailed in Subsections 4.2-4.5.

4.2. Heuristic solution

Heuristic FSIC was proposed in (Dolgui et al., 2005). Here, we suggest a modification of this heuristic. We adapt it for the case where operations are given by their parameters (feed per minute and working stroke) and not by their processing times. The heuristic tries to assign operations at random while respecting all given constraints. It makes $TR_{tot}$ iterations at maximum and returns the best assignment as the result. The operation to be assigned is chosen at random from a list of candidates. An operation is a candidate to be assigned if all its predecessors have been assigned and its assignment does not follow to violating the cycle time constraint. The operations that must be located at the same station have a priority. The general scheme of the modified FSIC heuristic is represented by Algorithm 2.

Algorithm 2.

Step 0. Set $C_{min} = \infty$, $TR_{tot} = 0$, $TR_{nimp} = 0$, seed=0.

Step 1. Set $m = 1$, $C_{cur} = C_1$.

while $T_{cur} < T_{res}$ and $TR_{nimp} < TR_{nimp,aut}$ and $TR_{tot} < TR_{tot,aut}$

Step 2. Build list $L_1$ of operations that can be assigned to the current station as follows:

Operation $i$ is added to $L_1$ if all following conditions are satisfied:

- for all $e \in ES$ if $i \in e$ then $e \cap (N_m \cup \{i\}) \neq e$;
- either $\exists l$ in $\{1, \ldots, n_m\}$ such that $t'(N_m) \leq T_0$ for $N_{ml} = N_{ml} \cup \{i\}$, or $t'(N_m) + \lambda/j + \tau b \leq T_0$ and $n_m + 1 \leq n_0$;
- if $(j, i) \in D$ then $j \in \bigcup_{r=1}^{m} \bigcup_{l=1}^{n_r} N_r$.
Step 3. If $L_1 \neq \emptyset$ then choose an operation $i$ randomly from list $L_1$, set list $L_2 = \{i\}$, 
else go to Step 7.

Step 4. If $\exists \ e \in ES$ such that $i \in e$, then add to list $L_2$ all operations from $e$ and all their non-assigned predecessors.

Step 5. Try to assign each operation from list $L_2$ to current station $m$ verifying all the constraints.

If it is possible, then go to Step 7.

Step 6. Set $C_{cur} = C_{cur} + C_1 + C_2n_m$, $m = m + 1$.

If $k > m_0$, then set $C=\infty$ and go to Step 10,
else try to assign each operation from list $L_2$ to new station $m$ verifying all the constraints.

If it is not possible, then set $C=\infty$ and go to Step 10.

Step 7. If all the operations of $N$ have been assigned, then go to Step 10.

Step 8. Build list $L_1$. If $L_1 \neq \emptyset$ then go to Step 2.

Step 9. Set $C_{cur} = C_{cur} + C_1 + C_2n_m$, $m = m + 1$.

If $k > m_0$, then set $C=\infty$ and go to Step 10, else go to Step 2.

Step 10. If $C_{min} > C_{cur}$ then set $C_{min} = C_{cur}$, $TR_{nimp} = 0$, $S_{min} = S_{cur}$, else set $TR_{nimp} = TR_{nimp} + 1$.

Step 11. Set $TR_{tot} = TR_{tot} + 1$.

end while

We use the decomposition of the initial problem into several sub-problems in order to reduce the solving time by decreasing the number of stations to which an operation can be assigned. In fact an operation belonging to $N^r$ can be assigned to one of $k^r$ stations but it can not be assigned to other stations because the sub-problems are solved independently. As a consequence, a global optimum is not reachable from all heuristic solutions. Exploring all initial solutions is impossible because of limited computational time. To reach high-quality solutions, an intelligent selection of initial solutions is needed. We use the following rule: try to avoid non promising solutions for the application of the decomposition. Since a solution is decomposed to a number of sub-problems, each of them consists of operations assigned to a number of consecutive stations. Therefore, more stations are concluded into a feasible solution more sub-problems we can obtain after the decomposition. It is obviously that the optimal solution of each sub-problem consists at least of one station. Thus, less stations we have in the
heuristic solution more chances we have to obtain a good final solution.

We introduce a variable $\delta$, which is used for the selection process. Let $C_{\text{heur min}}$ be the cost of the best solution obtained by the heuristic, $C_{\text{heur}}$ be the cost of the current heuristic solution. A heuristic solution will be chosen, if $C_{\text{heur}} < (1 + \delta) C_{\text{heur min}}$, otherwise we will not decompose such a solution and will start FSIC heuristic to obtain other one. This avoids the processing of unpromising solutions having a little chance to give a final solution of good quality.

**4.3. Decomposition of a heuristic solution**

The size of each subsequence (number of stations) $k'$ is chosen at random because we do not know a priori which subsequence of decomposition can be the most advantageous. Since $N^r$, the set of operations assigned to the $k'$ stations of subsequence $r$, defines the size of sub-problem $SP_r$, we have to avoid the following two extreme situations: i) if the size of a sub-problem is “too small”, the resolution by shortest path approach does not improve the heuristic solution; ii) if the size of a sub-problem is “too large”, it will be impossible to apply the shortest path approach for its resolution because of limited available time (the calculation time increases drastically with the problem size).

It is necessary to create sub-problems for which the shortest path approach is applicable. We introduce the following variables, which we use to limit the size of the sub-problems: i) maximal number of stations within one subsequence, $\text{max_st_sub}$ that is the maximum of $k'$; ii) maximal number of operations within one subsequence $\text{max_op_sub}$ that is the maximum of $|N^r|$. The value $k'$ is chosen at random within $[1, \text{max_st_sub}]$ and then can be modified so that the total number of operations in the sub-problem $N^r$ does not exceed $\text{max_op_sub}$ and the total sum of $k'$ is not greater than $m_{\text{heur}}$. Algorithm 3 shows how $k'$ and $N^r$ are determined.

**Algorithm 3.**
\[ k = \sum_{i=1}^{i-1} k^i; \]

\[ k' = \text{random}[1 .. \text{max}_\text{st}_\text{sub}]; \]

If \( k' + k > m_{\text{heur}} \) then \( k' = m_{\text{heur}} - k \)

\[ N^r = N_k, i=1 \]

while \( i < k' \) do

  if \(|N^r| + |N_{k+i}| < \text{max}_\text{op}_\text{sub}\) then \( N^r = N^r \cup N_{k+i}, i = i +1\), else break

end while

\[ k' = i \]

### 4.4. Resolution of sub-problems

Each subsequence \( r \) of \( k' \) stations is used to generate a new sub-problem \( SP_r \), which is of the same type as the initial problem but with a reduced size, where:

- The set of operations \( N \) is reduced to \( N^r \subset N \);
- Constraints (2) – (8) are transformed by replacing the set \( N \) with the subset \( N^r \), and then removing the constraints related to operations from \( N \setminus N^r \).
- Since an optimal solution can not have a cost greater than a feasible solution (i.e. \( C_{r_{\text{opt}}} \leq C_{r_{\text{heur}}} \)) and taking into account that a station always contains at least one block, we can decrease \( m_0 \) for the sub-problem \( SP_r \) down to \(|C_{r_{\text{heur}}} / (C_1+C_2)|\) where \( C_{r_{\text{heur}}} \) is the cost of subsequence \( r \) in the sequence \( S_{\text{heur}} \).

Then, each sub-problem \( SP_r \) is solved by the shortest path approach. Since, the optimal solution of the sub-problem \( C_{r_{\text{opt}}} \) is found, the algorithm replaces the old sequence of \( k' \) stations by the obtained optimal sequence. Therefore, we can decrease the number of stations and blocks for assignment of each set \( N^r \) and thus the total amount of equipment is decreased.

Since we choose an integral number of consecutive stations for creating a new sub-problem, this ensures that the initial incompatibility constrains between the operations of different sub-problems...
remain respected, therefore, all sub-problems can be solved independently. In order to respect the initial precedence constrains between the operations of different sub-problems, we only have to replace old subsequences by optimal ones in the same order that they were created.

The shortest path method was initially suggested in (Dolgui et al., 2000), for more recent version see (Dolgui et al., 2007). Let \( P \) be a set of collections \( P=\{\{N_{11},...,N_{1n_1}\},...,\{N_{m1},...,N_{mn_m}\}\} \), satisfying constraints (2)-(10). The set \( v_{kl} = \bigcup_{r=1}^{k-1} \bigcup_{q=1}^{n_r} N_{rq} \bigcup \bigcup_{q=1}^{l} N_{kq} \) is the set of operations executed on the part after machining it by the \( l \)-th block of the \( k \)-th workstation. Let \( V \) be the set of all \( v_{kl} \) for all \( P \in \mathbf{P} \), including \( v_0 = \emptyset \). We construct a digraph \( H = (V, F) \), in which there is an arc \((v', v'')\) in \( F \) iff \( v' \subset v'' \), and \( e \subset N'' = v'' \setminus v' \) for all \( e \in \overline{ESB} = \overline{EB} \cup \overline{ES} \), i.e. the arc \((v', v'')\) represents the block \( v'' \setminus v' \) of operations. For each arc \((v', v'')\), we assign a time parameter \( t(v', v'') = \hat{t}(N'') \). To each design decision \( P \in \mathbf{P} \) we associate a parameterized path \( x(P) = ((u_0 = v_0, ..., u_{j-1}, u_j, ..., u_{k(x)} = v_N), (\gamma_0 = 1, ..., \gamma_j, ..., \gamma_{l(x)} = 1)) \) in the digraph \( H \) from vertex \( v_0 \) to vertex \( v_N \). The parameter \( \gamma_j \) is equal to 1 if the block \( u_j \setminus u_{j-1} \) is the last block of the corresponding station of \( P \). The cost of a path \( x \) is defined by \( Q(x) = C_1 m(x) + C_2 l(x) \), where \( l(x) \) is the number of arcs in the path \( x \) and \( m(x) \) is the number of arcs with \( \gamma_j = 1 \). In this case, \( l(x) \) and \( m(x) \) correspond to the number of blocks and stations in the decision \( P \), respectively.

The algorithm developed in (Dolgui et al., 2000, 2007) finds a shortest constrained path in digraph \( H \), i.e. a path \( x_{opt} \) with the minimal cost such that \( m(x) \leq m_0 \) and \( j_k \leq j_{k-1} \leq m_0 \) for \( k = 1, ..., m(x) \). The solving time depends on two parameters of the problem: the cardinality of the set of operations to be assigned and the number of constraints. Obviously, a greater number of operations leads to increasing the size of \( H \) because of increasing the set of all possible \( v_{kl} \). In contrast, constraints reduce the number of possible \( v_{kl} \) and thus decrease the size of \( H \). Tests showed that the shortest path method is applicable for small and medium-sized problems (less than 60 operations).
If it is impossible to avoid a sub-problem of “too small” size (for example, at the end of decomposition when only one station remains). A sub-problem has “too small” size if it was formed on the basis of a subsequence consisting of only one station with the number of blocks less than three or of two stations with the number of blocks less than two each. In such a situation, the shortest path method is not used and the old sequence (obtained by the heuristic algorithm) of stations is kept.

A sub-problem of large size also can be obtained (for example, a subsequence consists of only one station, but the number of operations assigned to this station exceeds $max_{op_{sub}}$), then it is necessary to decide to solve or not this sub-problem exactly. Decision depends on the estimated time required for solving sub-problem. In order to estimate it, two parameters of a sub-problem are analyzed: the cardinality of the set of operations to be assigned and the number of constraints. If the estimated time is long (in comparison with the total available time $T_{res}$), such sub-problem is not solved with the shortest path method and the old sequence of stations (obtained by the heuristic algorithm) is kept.

4.5. Formation of a final solution

The total number of sub-problems $k_{num}$ of decomposition is not known in advance, because values $k'$ are chosen at random. The resolution of all sub-problems $SP_r, r = 1, .., k_{num}$ allows getting a new feasible solution $P_{hybr}$ of the initial problem that consists of $m_{hybr}$ stations. It is obviously that the cost of this solution satisfies $C_{hybr} \leq C_{heur}$. It is easy to form a final solution on the basis of the partial solutions because it is enough to connect the solutions of the sub-problems $SP_r$ (or saved heuristic partial solutions if the corresponding sub-problems were not solved) in the same order that sub-problems were created to obtain a feasible solution of the initial problem with cost $C_{hybr}$ which is the sum of costs of partial solutions.
5. Experimental study

The purpose of this study is to compare the proposed approach with FSIC heuristic (see Subsection 4.2). The results of 14 series of 10 test instances each are reported in Table 1. They were generated randomly for different values of $|N|$ and order strength ($OS$). The order strength is defined as the density of the transitive closure of the precedence graph (Scholl and Klein, 1998). All methods were implemented using C++, the experiments were carried out on Pentium IV, 3GHz with 512 Mb of RAM.

Each series was solved by FSIC heuristic and this heuristic multi-start decomposition algorithm with constant parameters $max_{st_{sub}} = 3$, $max_{op_{sub}} = 30$, $TR_{tot} = 10000$, $TR_{nimp} = 10000$ and with different values of parameter $\delta = \{0.01, 0.02, 0.025, 0.03, 0.04, 0.05\}$. Both approaches were compared for different resolution times $T_{res}$.

For the presentation of the obtained results we use the following notations:

$\Delta_{max}, \Delta_{av}, \Delta_{min}$ – maximal, average and minimal deviation of the line cost obtained by the multi-start decomposition approach from the cost obtained by the heuristic, respectively.

$\delta_{opt}$ – the value of parameter $\delta$ for which the maximum of $\Delta_{av}$ was reached.

[Insert Table 1 about here]

The obtained results show that:

- For all tests the average deviation of the line cost obtained by the multi-start decomposition approach from the cost obtained by the heuristic FSIC is negative;
- For all tests the minimal deviation of the line cost obtained by the multi-start decomposition approach from the cost obtained by the heuristic is negative or equal to 0;
- The maximum of the attained improvement is greater than 10%.

The optimal value of the parameter $\delta$ is within $[0.02, 0.03]$. 

19
6. Conclusion

We have considered a crucial industrial problem for machining lines the Transfer Line Balancing Problem (TLBP). In our previous work, for this problem, we suggested some exact optimization methods and heuristics. In this paper, we proposed a new multi-start decomposition algorithm combined the shortest path method (exact algorithm) and the FSIC method (heuristic). The multi-start decomposition algorithm uses the heuristic to obtain initial solutions. Such a solution is then broken down into several sub-problems. For each sub-problem we apply the shortest path approach to improve the partial solution. The procedure is repeated several times until the available computation time is exceeded.

The proposed multi-start decomposition algorithm is relatively efficient for large-scale problems. This conclusion is based on our experimental comparison of this algorithm with the FSIC heuristic which is the best known heuristic for TLBP problem. These experiments show that the quality of the solutions obtained by the multi-start decomposition approach depends on the characteristics of the problem (number of operations, order strength) as well on control parameters and available calculation time.

The developed method can be applied with other heuristics and exact approaches. The quality of the obtained results will depend certainly on the quality of algorithms used.

Acknowledgments: This research was financially supported by INTAS under Project 03-51-5501.

References


Figure 1. Scheme of the multi-start decomposition algorithm

Solution $P_{heur}$ is presented by a sequence of $m_{heur}$ stations.

Solution $P_{heur}$ of initial problem consists of $m_{heur}$ stations, $m_{heur} \leq m_{heur}$. 
<table>
<thead>
<tr>
<th>N</th>
<th>OS, %</th>
<th>T&lt;sub&gt;res&lt;/sub&gt;, min</th>
<th>δ&lt;sub&gt;opt&lt;/sub&gt;</th>
<th>Δ&lt;sub&gt;max&lt;/sub&gt;, %</th>
<th>Δ&lt;sub&gt;av&lt;/sub&gt;, %</th>
<th>Δ&lt;sub&gt;min&lt;/sub&gt;, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>90</td>
<td>30</td>
<td>0.02-0.03</td>
<td>-1.88</td>
<td>-0.65</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>30</td>
<td>0.02-0.03</td>
<td>-3.02</td>
<td>-0.84</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>30</td>
<td>0.02-0.03</td>
<td>-2.12</td>
<td>-0.81</td>
<td>0</td>
</tr>
<tr>
<td>125</td>
<td>90</td>
<td>30</td>
<td>0.025</td>
<td>-7.24</td>
<td>-3.80</td>
<td>-0.86</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>30</td>
<td>0.02</td>
<td>-4.99</td>
<td>-1.06</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>30</td>
<td>0.02</td>
<td>-10.9</td>
<td>-2.2</td>
<td>0</td>
</tr>
<tr>
<td>150</td>
<td>90</td>
<td>30</td>
<td>0.03</td>
<td>-7.06</td>
<td>-3.83</td>
<td>-1.94</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>30</td>
<td>0.03</td>
<td>-9.86</td>
<td>-4.08</td>
<td>-2.6</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>30</td>
<td>0.03</td>
<td>-9.75</td>
<td>-6.37</td>
<td>0</td>
</tr>
<tr>
<td>200</td>
<td>90</td>
<td>60</td>
<td>0.03</td>
<td>-6.6</td>
<td>-4.63</td>
<td>-2.09</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>60</td>
<td>0.03</td>
<td>-5.1</td>
<td>-2.69</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>60</td>
<td>0.03</td>
<td>-6.58</td>
<td>-3.24</td>
<td>0</td>
</tr>
<tr>
<td>250</td>
<td>90</td>
<td>120</td>
<td>0.03</td>
<td>-6.79</td>
<td>-5.2</td>
<td>-3.28</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>120</td>
<td>0.03</td>
<td>-5.44</td>
<td>-3.11</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1. Results of tests