

Some new ideas for assembly line balancing research

Alexandre Dolgui* Evgeny Gafarov**

* *Ecole des Mines de Nantes, IRCCYN - UMR CNRS 6597, La Chantrerie, 4, rue Alfred Kastler - B.P. 20722, F-44307 Nantes cedex 3, France (email: alexandre.dolgui@mines-nantes.fr)*

** *V.A. Trapeznikov Institute of Control Sciences of the Russian Academy of Sciences, Profsoyuznaya st. 65, 117997 Moscow, Russia. Tel.: +7 925 809 09 07. (email: axel73@mail.ru)*

Abstract: The design of assembly lines is an important issue in manufacturing engineering, management and control. The idle time is the most interesting performance index for assembly line design. The classical simple line-balancing problem (SALBP) consists of assigning tasks, necessary for production of a product, to workstations such that the idle time (number of stations, cycle time, cost) is minimized while precedence constraints between tasks are satisfied. From the worst-case analysis point of view, the SALBP problems are NP-hard in strong sense. Nevertheless, in practice, it is also important to be able to compare real instances of SALBP. In this paper, the simple assembly line balancing problem of type 1 (SALBP-1) is considered where the cycle time is fixed and the objective is to minimize the number of stations. Two unconventional ways are proposed to help to estimate the complexity of such a problem instances: reduction of the graph of precedence constraints to planar one and transformation of the problem to a problem of maximization. We show how these techniques can be employed and why they are useful to analysis of assembly line balancing problem instances.

Keywords: Modeling of assembly units, Modeling of manufacturing operations, Job and activity scheduling, Assembly line balancing, Algorithms, Graphs

1. INTRODUCTION

Modern production systems are characterized by short life-cycle times of products, high level of automation, emergence of new manufacturing equipment and technologies, and high investment.

Assembly is a key production activity, to put in place such an activity it is often necessary to develop assembly lines. Assembly lines are production systems that include serially located workstations in which operations (tasks) are continuously processed. They are employed in various industries like the automobile, electronics, etc. where the objective is to produce large series of products, see Dolgui and Proth (2010).

The design of assembly lines is an important problem in manufacturing engineering, management and control, Askin and Standridge (1993). The stations workloads balancing is the most interesting performance index for assembly line design. The classical simple line-balancing problem (SALBP) consists of assigning tasks, necessary for production of a product, to workstations such that the idle time (or number of stations, cycle time, cost) is minimized and precedence constraints between tasks are satisfied. The SALBP problems are NP-hard. Thus, in real life applications, it is crucial to know if it is possible to solve this problem for a given number of tasks. What is why the possibility to evaluate the complexity of a SALBP instance is an important issue.

In this paper, the simple assembly line balancing problem of type 1 (SALBP-1) is considered where the cycle time is fixed and the objective is to minimize the number of stations, see Baybars (1986). Two unconventional ways are proposed to estimate the complexity of such a problem instances and to improve algorithms to solve the problem: modification of the graph of precedence relations and investigation of a problem with an opposite optimality criteria. We show how these techniques can be employed and why they are useful to analysis of assembly line balancing problem instances and to increase the performance of existing methods.

Surveys on assembly line balancing techniques are presented in Scholl and Becker (2006), interesting classification and representation schemes for line balancing problems are proposed in Battaia and Dolgui (2013).

Salveson (1955) was the first to empirically address the SALB problem. He described it as a set of precedence constraints between tasks that must be adhered to. Salveson formulated the SALBP as a linear programming problem including all possible combination of station assignments. His model, by definition, can result in splitting tasks and may generate infeasible solutions. Bowman (1960) was the first to provide a 'non-divisibility' constraint, by changing the linear programming formulation to a zero-one integer programming.

The SALBP can be also represented by a tree in which each path corresponds to a feasible solution where each arc represents a station. Jackson (1956) was the first to propose an algorithm for SALBP-1 using the notion of a tree. A station is termed maximal if no task can be assigned to it without violating the precedence and the cycle time constraints. The method starts by generating all feasible assignments to the first station (one of these feasible assignments will obviously be part of the optimal solution). Then it generates all feasible assignments to the second station, given the first station assignments. Then for the first-second station combination, all feasible solutions are constructed for the third station, etc.

The process is repeated, each time adding one station. Jackson uses dominance arguments to eliminate inferior feasible assignments: after generating all feasible assignments, these sets are compared to eliminate subsets that contain exactly the same tasks as other subsets and to eliminate the dominated subsets. A set is said to be dominated if it contains fewer tasks than another subset and does not contain any tasks that are not also in the other subset. Jackson's method does not require the generation of all feasible sequences, but still uses exhaustive search of all remaining possibilities.

Agnetis and Arbib (1997) addressed the problem of assigning and sequencing tasks to stations with the objective of minimizing the inventory costs or the completion time of a set of identical jobs. They consider two assignment policies: the standard assignment (tasks and their pre-processing tasks are grouped on the same station) and the look-ahead assignment (tasks and their pre-processing tasks, are not necessarily grouped on the same station). Three types of sequencing policies are proposed where the main difference is the order in which the tasks assigned to a station are executed. The method is limited to a single product and it is based on the dynamic programming technique.

Since the ALB problem falls into the NP-hard class of combinatorial optimization problems, numerous research efforts have been directed towards the development of computer efficient approximation or heuristics algorithms Ghosh and Gagnon (1989).

Wee and Magazine (1982) proposed to solve the problem by means of the generalized bin packing heuristics. A Branch and Bound method similar to the one used for the bin-packing problem is used. The authors also proposed techniques used in the bin packing problem like the first-fit-decreasing (FFD) rule and the immediate-update FFD (IUFFD) to the ALBP. In the FFD rule method, the tasks are listed in a non-increasing order of processing time. In the case of IUFFD, the set of available tasks is immediately updated and arranged in non-increasing order of processing time. In general, these two heuristics give good results.

One of the first proposed heuristic was the ranked positional weight (RPW) Helgeson and Birnie (1961). The main idea behind the RPW heuristic is to first assign the tasks, which have long chains of succeeding tasks. The length of the chain can be measured either by the number of successor tasks, or by the sum of the task times of the successor tasks. In the RPW method, the sum of the task

processing time and the processing times of the successor tasks is defined as the positional weight of the task. The tasks are ranked in descending order of the positional weights with ties broken arbitrarily. The tasks are then picked in their ranked order and assigned to stations if (1) all predecessors of the task have already been assigned and (2) the task fits in the remaining time on the station. If a task does not fit in the remaining time on a station it is skipped and the next task in the ranked order is selected. If no task fits into the station, the station is closed and a new station is opened. The tasks are then scanned from the beginning of the list and an attempt is made to assign unassigned tasks to the new station. The process is repeated until all tasks are assigned. Since the weight of an element depends on the processing time of its predecessors, the algorithm tends to treat the elements near the end of precedence graph first. This procedure is quite simple it can deal with user's preferences but gives good results only for simple problems.

The 'reversed ranked positional weight' (RRPW) is the RPW method applied to the problem with the reversed precedence constraints. After a balance is found for the reversed problem, the problem is 'unreversed' to get a balance solution for the original problem.

Kilbridge and Wester (1961) proposed a heuristic based on the cumulative performance times. First the 'layers' are identified in the precedence graph. Tasks with no predecessors are in the first layer. Tasks that are preceded directly by tasks in layer i are placed in layer $i+1$, etc. Then, it computes the time for each layer (sum of processing time of tasks belonging to a given layer), as well as the cumulative performance time of each layer (sum of its time and times of the layers it succeeds). Moving from the left to the right in units of a layer, it finds the layer that just fills or overfills the station. It begins by the first layer and goes in order through the remaining layers. Tasks of a given layer are assigned to a given station unless the precedence constraints will be violated or the cycle time will be exceeded, otherwise they are assigned to the next station and so on. The method is a search algorithm, which is restricted to looking one layer behind and one layer ahead in filling the stations. This heuristic is easy to implement and gives good solutions for dense and uniform graphs. The more the graph contains sprays, less the method yields good balancing.

Arcus (1966) proposed the heuristic COMSOAL (COMputer Method for Sequencing Operations for Assembly Lines). The basic idea: this algorithm randomly generates a large number (e.g. 1000) of feasible solutions and selects the best one with minimal station number. A solution is constructed by assigning operations to stations, starting from the first station. The next operation to be assigned is selected randomly from the subset F of such operations, for which: processing time inferior the idle time remaining for current station; all predecessors assigned already. If the subset F is empty, the next workstation is open and so on.

On the evolutionary algorithm side, to our knowledge, the first attempt was by Falkenauer and Delchambre (1992) who used a grouping genetic algorithm (GGA). The authors generalised their bin packing GGA to obtain a fast algorithm supplying high-quality approximated solutions

of the ALB problem. One advantage of their method was its ability to handle problems with sparse, even empty precedence constraints, thanks to its bin packing 'ancestor'. Another advantage lies with the fact that the GA is a gradual improvement method, thus giving the user the possibility when the computational resources allow it to extend the search and to continue to improve the currently available solution. Note that the mechanism of complying with precedence constraints applies equally well to the hybrid GGA of Falkenauer (1996).

Shtub and El (1990) pointed out the gap existing between literature dealing with the ALB problem and technical documentation used by engineers to describe assembly operations. Comparing the precedence graph to the assembly chart, it is obvious that precedence graph, which presents tasks by their times and precedence relations does not fully describes the relations between a task and the subassemblies on which this task is performed. An ALB method based only on precedence graph is likely to generate solutions that minimize idle time by loading each station with tasks performed on several subassemblies. Such solutions are in contradiction with the very basic principles of improving work methods and enriching employee jobs. The author proposed two ways to deal with the relationship between a task and the subassemblies on which the task is performed. The first approach aims to minimize the number of subassemblies handled by each station. The second approach aims to constraint the number of sub-assemblies by each station

The Simple Assembly Lines Balancing Problem of type 1 (SALBP-1), considered in this paper, is formulated as follows:

A set $N = \{1, 2, \dots, n\}$ of operations is given. For each operation $j \in N$, a processing time $t_j \geq 0$ is known. A cycle time $c \geq \max\{t_j, j \in N\}$ is also known and fixed. Furthermore, the finish-start precedence relations $i \rightarrow j$ are defined between the operations according to an acyclic directed graph G .

The objective is to assign each operation j , $j = 1, 2, \dots, n$, to a station in such a way that:

- number m of stations used is minimized;
- for each station $k = 1, 2, \dots, m$, its total load time $\sum_{j \in N_k} t_j$ does not exceed c , where N_k is the set of operations assigned to station k ;
- precedence relations are fulfilled, i.e. if $i \rightarrow j$, $i \in N_{k_1}$ and $j \in N_{k_2}$, then $k_1 \leq k_2$.

A fundamental and comprehensive analysis of SALBP was done in Baybars (1986). The most recent algorithm for SALBP-1 is presented in Morrison et al. (2014).

It is known that from the worst case analysis point of view, the SALBP-1 is NP-hard in the strong sense. The literature on SALBP-1 is rich. Nevertheless, some specific studies on comparison of SALBP-1 instances is missing. Indeed, it will be useful to suggest techniques helping to evaluate a relative complexity of real life instances and academic benchmarks.

From this perspective, here two new ways for simple assembly line balancing research are suggested: i) reduction of precedence graphs to planar ones to minimize

the number of precedence relations have to be considered and ii) transformation of the SALBP-1 to a maximization problem to estimate its solution space. These approaches open completely new and promising ways for comparison of SALBP instances. They can be also a departure point to develop new optimization algorithms.

In Section 2, a new idea dealing with transformation of precedence graphs to planar ones is discussed. Examples of such transformations are given. This approach can reduce the complexity of specific examples of simple assembly line balancing problems as well as can be used to compare the known benchmark instances employed to test different optimization algorithms.

In Section 3, a new simple assembly line balancing problem where the number of stations is maximized, is suggested. Only solutions with maximal station loads are considered. Some examples of solving such a problem are reported. This transformation of SALBP-1 to a maximization problem can be used to evaluate the complexity of simple assembly line balancing problem instances.

2. PLANAR GRAPH FOR PRECEDENCE RELATIONS

In literature, *order strength* is considered as an important characteristic of SALBP benchmarks A. (1993). On <http://www.assembly-line-balancing.de>, order strength is estimated according to number $n \cdot (n - 1) / 2$ of precedence relations. Thus, the number of precedence relations is an important measure of SALBP instance complexity. In this section, it is demonstrated how to reduce graph of precedence to planar one as well as benefits of such a reduction in term of number of precedence relations.

Here it will be proved that any graph can be reduced to planar one with a number of precedence relations no greater than $3n - 6$. If in the original graph, operation i is a predecessors of operation j (immediate or not), than this relation remains in the modified planar graph too. Thus, we can suggest modifying a graph before solving an instance because this modification reduce the order strength of the problem and so its complexity.

In Lazarev and Gafarov (2009), authors present a transformation of graph G of precedences to planar one for the Resource-Constrained Project Scheduling Problem (RCPSP). The same approach can be used for SALBP-1.

Theorem 1. For any instance of SALBP-1 with n operations and v precedence relations, there exists an analogous instance with a planar graph G' with n' operations and v' relations, where $n + v \geq n' + v'$.

An analogous instance can be obtained from the original by adding "dummy" operations (with $t_j = 0$) and deleting all unnecessary relations. The proof of the theorem follows from Lemmas 6 and 3.

Lemma 2. If there is a subgraph $G' \subset G$ that is isomorphic to the special graph $K_{3,3}$ (complete bipartite graph on six vertices, three of which connects to each of the other three, also known as the utility graph), then it is possible to transform it into a planar subgraph by adding "dummy" jobs (with $t_j = p = 0$) and deleting all the unnecessary

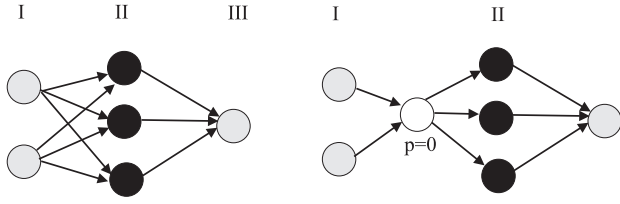


Fig. 1. Transformation of $K_{3,3}$

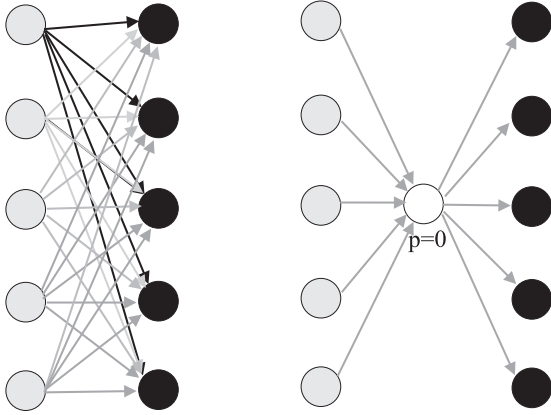


Fig. 2. Transformation of $K_{k,k}$

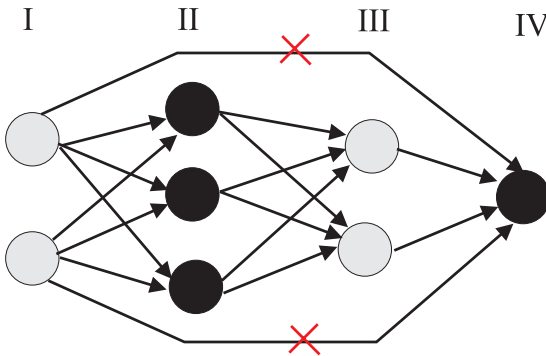


Fig. 3. Transformation of $K_{4,4}$

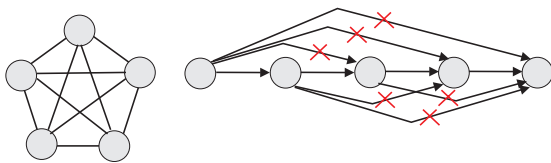


Fig. 4. Transformation of $K_{5,2}$

relations. See the transformation rules shown in Figures 1, 2 and 3.

Lemma 3. If there is a subgraph $G' \subset G$ that is isomorphic to the special graph $K_{5,2}$ (the complete graph on five vertices), then it can be transformed into a planar subgraph by deleting all the unnecessary relations. See the transformation rule shown in Figure 4.

Thus, according to the Theorem of Pontryagin and Kuratowski Kuratowski (1930), the modified graph is planar. According to Euler's Theorem, in such a planar graph $v' \leq 3n' - 6$.

The number of precedence relations influences the running time and theoretical complexity of algorithms. Usually,

III the time complexity of solution algorithms for SALBP-1 looks like $O(f(n + v))$, i.e. depends almost equally on n and v . For example, in a list scheduling algorithm, after assigning an operation, it is necessary to consider all arcs directed from it, to check if its successors are available for the value $n + v$ is reduced, then running time will be reduced too. Of course we add dummy jobs, but algorithms can be adapted in a way that dummy jobs will not affect running time substantially, because it is known which jobs are dummy.

Moreover, by transforming graphs of precedence relations for benchmarks to planar ones, the benchmarks will be normalized, and thus it will be possible to compare all algorithms in the exactly same conditions.

3. MAXIMIZATION OF NUMBER OF STATIONS ON SOLUTIONS WITH MAXIMAL STATION LOADS

In this section, instead of the standard SALBP-1, where the number of stations is minimized, the simple assembly line balancing with the opposite objective criterion is considered: the number of stations is maximized. Similar approaches are known in the scheduling theory Aloulou et al. (2004).

To make this maximization problem non trivial, it is assumed that all stations should have a maximal workload, i.e. for two stations m_1 and m_2 ($m_1 < m_2$), there is no operation j assigned to station m_2 which can be moved to station m_1 without violating precedence or cycle time constraints.

Let m be the number of stations for a feasible solution with maximal station loads, m^{min} be the minimal number and m^{max} be the maximal number of stations for solutions with maximal station workloads.

Denote maximization simple assembly line balancing problem as max-SALBP-1. In the following, it will be proved that the max-SALBP-1 is NP-hard in the strong sense.

3-Partition problem:

A set $N = \{b_1, b_2, \dots, b_n\}$ of $n = 3\bar{m}$ positive integers is given, where $\sum_{i=1}^n b_j = \bar{m}B$ and $\frac{B}{4} < b_j < \frac{B}{2}$, $j = 1, 2, \dots, n$. Does there exist a partition of N into \bar{m} subsets $\bar{N}_1, \bar{N}_2, \dots, \bar{N}_{\bar{m}}$ such that each subset consists exactly of three numbers and the sum of the numbers in each subset is the same, i.e.,

$$\sum_{b_j \in \bar{N}_1} b_j = \sum_{b_j \in \bar{N}_2} b_j = \dots = \sum_{b_j \in \bar{N}_{\bar{m}}} b_j = B?$$

Lemma 4. max-SALBP-1 is NP-hard in the strong sense.

Proof. Use a reduction from the 3-Partition problem. Consider an instance of the 3-Partition problem with $3\bar{m}$ numbers. Let $M = (\bar{m}B)^2$. Construct an instance of max-SALBP-1 with $3\bar{m} + 1$ operations, where $t_j = M + b_j$, $j = 1, 2, \dots, 3\bar{m}$ and $t_{3\bar{m}+1} = M$. In addition, let $c = B + 4M - 1$.

If for the instance of the 3-Partition problem the answer is "YES", then there exists an optimal solution with $m + 1$ stations. Operations which correspond to numbers from the set \bar{N}_i are assigned to the station i , $i = 1, 2, \dots, \bar{m}$. The operation $3\bar{m} + 1$ is assigned to the station $\bar{m} + 1$. If

the answer is "NO", then $m^{max} = \overline{m}$, and on a station, 4 operations are assigned (including the operation $3\overline{m} + 1$).

As a consequence the following lemma holds.

Lemma 5. max-SALBP-1 cannot be approximated with a ratio $< \frac{3}{2}$ unlike $P = NP$.

In Queyranne (1985), it was proven that for any polynomial time heuristic for SALBP-1, the worst-case ratio is at least $\frac{3}{2}$, i.e. for a heuristic algorithm there is an instance for which $\frac{m}{m^{min}} \geq \frac{3}{2}$.

SALBP-1 is a generalization of the Bin-Packing Problem (BP), thus this approximation can be compared with known results for the BP (e.g., see Queyranne (1985)), where there is a heuristic for which $m \leq 11/9m^{min} + 1$. For BP, it is known that for any polynomial time heuristic, the worst-case ratio is at least $\frac{3}{2}$, as well. But it holds only for $m^{min} = 2, m = 3$, i.e. the absolute error is equal to 1. For $m^{min} > 2$, the worst case is $m \leq 11/9m^{min} + 1$. It is possible to conjecture, that the same relation holds for SALBP-1 too, but in Queyranne (1985), authors showed that the worst-case ratio $\frac{3}{2}$ holds for any absolute error, i.e., for any polynomial time heuristic and for any $m^{min} \geq 2$, there is an instance for which $\frac{m}{m^{min}} = \frac{3}{2}$, where m is computed by the heuristic.

A similar result for max-SALBP-1 can be demonstrated.

Lemma 6. For any polynomial time heuristic for max-SALBP-1 and for any given absolute error $q \geq 3$, there is an instance for which $\frac{m^{max}}{m} = \frac{3}{2}$ and $m^{max} - m = q$.

Proof. Consider the following reduction from the Partition Problem. Let an instance of the Partition Problem with n values be studied. Without loss of generality assume $A = \frac{1}{2} \sum_{i=1}^n b_i$. Construct an instance of max-SALBP-1 with a set of operations $N = N_1 \cup N_2 \cup \dots \cup N_q$. Assume $c = 2n^2 A \cdot n^{2q}$. Each set N_l , $l = 1, 2, \dots, q$, contains $2n + 3$ operations, with processing times $t_j = M_l + b_j$, $j = 1, 2, \dots, n$, and $t_j = M_l$, $j = n + 1, \dots, 2n + 1$ and $t_{2n+2} = t_{2n+3} = T_l = c - ((n + 1)M_l + A - 1)$, where $M_q = 2n^2 A$ and $M_l = M_{l+1} \cdot (n + 2)$, $l = q - 1, q - 2, \dots, 1$. In addition, assume $i \rightarrow j$, $\forall i \in N_l$, $\forall j \in N_{l+1}$, $l = 1, 2, \dots, q - 1$ and $2n + 2 \rightarrow i$, where $2n + 2, \forall i \in N_l$, $l = 1, 2, \dots, q$.

Jobs from the different subsets N_l , $l = 1, 2, \dots, q$, cannot be assigned to the same station. Jobs $2n + 2$ and $2n + 3$ from the same subset are assigned to different stations too. If for the instance of the Partition problem the answer is "YES", then there exists an optimal solution with $m^{max} = 3q$ stations, where jobs from the subset N_l , $l = 1, 2, \dots, q$, are assigned to stations $3(q - 1) + 1$, $3(q - 1) + 2$, $3(q - 1) + 3$ according to their assignment from the proof of Lemma 4. For the instance in a line balance with $m^{min} = 2q$ stations, jobs from each subset N_l , $l = 1, 2, \dots, q$, occupy only 2 stations.

The lemma is proven.

The calculation results of maximal number of stations for several benchmarks from <http://www.assembly-line-balancing.de> are reported here. To maximize the number of stations, a simple B&B algorithm with the deep-first branching strategy and a simple Upper Bound based on the following observations were developed.

Let us compute $Pred_j$ which is a set of all predecessors (not immediate, as well) for each operation j , $j = 1, 2, \dots, n$. Then, the earliest station S_j to which the operation $j = 1, 2, \dots, n$, can be assigned is computed as $S_j = \lceil \frac{(\sum_{i \in Pred_j} t_i) + t_j}{c} \rceil$. Let ML_k be the minimal load of the station k , $k = 1, 2, \dots, n$. Let $t_k^{min} = \min_j \{t_j | S_j \leq k\}$ and $t_k^{max} = \max_j \{t_j | S_j \leq k\}$. Then, $ML_k = \max\{t_k^{min}, c - t_k^{max} + 1\}$ and $UB_1 = \min\{m | \sum t_j - \sum_{l=1}^m ML_l \leq 0\}$ is an Upper Bound for the maximal number of stations.

Some other Upper Bounds are used, e.g., $UB_2 = \lceil \frac{2 \cdot \sum t_j}{c} \rceil - 1$. Upper bounds are recomputed for each subproblem. Unfortunately, the B&B proposed cannot solve the majority of benchmarks in 10 minutes (600 seconds) on CPU INTEL CORE 2 DUO 2.4 Hz (only one processor is used). However, the results obtained, see Table 1, can be used to estimate the difference between the minimal and maximal numbers of stations for these benchmarks.

The purpose of this numerical tests was not to check the efficiency of the algorithm proposed but to give a view how big is difference between the maximal and minimal numbers of stations for considered benchmarks.

Table 1. Minimal and maximal numbers of stations for known benchmarks

Instance	n	c	m^{min}	m^{max}	running time (sec)
Arcus1	83	3786	21	24	600.00
Arcus2	111	5755	27	30	600.00
Barthol2	148	84	51	57	600.00
Barthold	148	403	14	16	600.00
Bowman	8	20	5	5	<0.01
Buxey	29	27	13	16	600.00
Gunther	35	41	14	16	600.00
Hahn	53	2004	8	9	600.00
Heskiaoff	28	138	8	10	600.00
Jackson	11	7	8	9	<0.01
Jaesche	9	6	8	8	<0.01
Kilbridge	45	56	10	12	600.00
Lutz1	32	1414	11	13	600.00
Lutz2	89	11	49	52	600.00
Lutz3	89	75	23	25	600.00
Mansoor	11	48	4	5	<0.01
Mertens	7	6	6	6	<0.01
Mitchell	21	14	8	10	0.36
Mukherje	94	176	25	27	20.90
Roszieg	25	14	10	11	247.32
Sawyer	30	25	14	17	600.00
Scholl	297	1394	50	54	600.00
Tonge	70	160	23	26	600.00
Warnecke	58	54	31	36	600.00
Wee-mag	75	28	63	63	14.66

If the running time less than 600, then the value m^{max} is optimal. After running the algorithm for 60 seconds, the same values m^{max} for all the instances was obtained, except for *Arcus2* (after 60 sec, $m^{max} = 29$) and *Lutz1* (after 60 sec, $m^{max} = 12$). These results show that the maximal deviation $m^{max} - m^{min}$ found does not exceed 20%.

4. CONCLUSION

In this paper, two new ideas for a future research on simple assembly line balancing of type 1 (SALBP-1) are

proposed: i) techniques to reduce graphs of precedence to planar ones; ii) the maximization version of SALBP-1, called max-SALBP-1.

The technique to reduce graphs of precedence to planar ones deals with a more precise calculation of complexity for benchmarks. All benchmarks can be normalized by applying such a reduction, thus only planar graphs of benchmarks will be compared and not any graphs. The perspectives for this research are in analyzing behavior of different known algorithms and differences in their performances on all known normalized benchmarks. An interesting topic for the future research is also the development of new line balancing algorithms based on the properties of planar graphs.

The maximization version of SALBP-1 provides new elements for understanding why some benchmarks harder to solve than others with greedy heuristics.

In this paper, it was demonstrated that this new problem max-SALBP-1, is NP hard in strong sense. A tentative algorithm to solve it is also proposed. Numerical tests are given for several known benchmarks. Further studies are necessary on all available benchmarks to understand better relations between the difference $max - min$ number of stations and problem complexity. Also, an open question is how this characteristic of problem can be used for the development of new and more efficient algorithms for SALBP-1.

ACKNOWLEDGEMENTS

The work was partially supported by the Saint-Etienne Métropole, France.

REFERENCES

- A., S. (1993). Data of assembly line balancing problem. *Technische Hochschule Darmstadt*, 16/93, 32 pp.
- Agnētis, A. and Arbib, C. (1997). Concurrent operations assignment and sequencing of particular assembly problems in flow lines. *Annals of Operations Research*, 69, 1 – 31.
- Aloulou, M., Kovalyov, M., and Portmann, M.C. (2004). Maximization problems in single machine scheduling. *Annals of Operations Research*, 129, 21 – 32.
- Arcus, A. (1966). Comsoal: A computer method of sequencing operations for assembly lines. *International Journal of Production Research*, 4, 259 – 277.
- Askin, G. and Standridge, C. (1993). *Modeling and Analysis of Manufacturing Systems*. John Wiley & Sons.
- Battaia, O. and Dolgui, A. (2013). A taxonomy of line balancing problems and their solution approaches. *International Journal of Production Economics*, 142(2), 259 – 277.
- Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing. *Management Science*, 32, 909–932.
- Bowman, E. (1960). Assembly line balancing by linear programming. *Operations Research*, 8 (3), 385 – 389.
- Dolgui, A. and Proth, J.M. (2010). *Supply Chain Engineering: Useful methods and techniques*. Springer.
- Falkenauer, E. (1996). A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, 2 (1), 5 – 30.
- Falkenauer, E. and Delchambre, A. (1992). A genetic algorithm for bin packing and line balancing. *Proceedings of the IEEE International Conference on Robotics and Automation*, 1186 – 1192.
- Ghosh, S. and Gagnon, R.J. (1989). A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. *Int. J. of Prod. Res.*, 27, 637 – 670.
- Helgeson, W.B. and Birnie, D.P. (1961). Assembly line balancing using the ranked positional weight technique. *J. Ind. Engng*, 12 (6), 394 – 398.
- Jackson, J. (1956). A computing procedure for the line balancing problem. *Management Science*, 2, 261 – 271.
- Kilbridge, M. and Wester, L. (1961). A heuristic method for assembly line balancing. *Journal of Industrial Engineering*, 12, 292 – 298.
- Kuratowski, K. (1930). Sur le probleme des courbes gauches en topologie. *Fundamental Mathematics*, 15, 271–283.
- Lazarev, A. and Gafarov, E. (2009). Transformation of the network graph of scheduling problems with precedence constraints to a planar graph. *Doklady Mathematics*, 79/1, 1–3.
- Morrison, D., Sewell, E., and Jacobson, S. (2014). An application of the branch, bound, and remember algorithm to a new simple assembly line balancing dataset. *European Journal of Operational Research*, 236/2, 403–409.
- Queyranne, M. (1985). Bounds for assembly line balancing heuristics. *Operations Research*, 33, 1353–1359.
- Salveson, M. (1955). The assembly line balancing problem. *J. Ind. Engng.*, 6 (3), 18 – 25.
- Scholl, A. and Becker, C. (2006). State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, 168(3), 666 – 693.
- Shtub, A. and El, E.D. (1990). An assembly chart oriented assembly line balancing approach. *Int.J. of Prod. Res.*, 28 (6), 1137 – 1151.
- Wee, T. and Magazine, M. (1982). Assembly line balancing as generalized bin packing. *Operations Research Letters*, 1, 56 – 58.