A Top-Down Approach for an Automatic Precedence Graph Construction under the Influence of High Product Variety

Simon Altemeier, Daniel Brodkorb, and Wilhelm Dangelmaier

Heinz Nixdorf Institute, University of Paderborn,
Fürstenallee 11, 33102 Paderborn, Germany
{Simon.Altemeier,daniel.brodkorb,wilhelm.dangelmaier}@hni.upb.de
http://www.hni.upb.de/cim

Abstract. This paper describes a top-down method for an automatic precedence graph construction that can cope with high variant products. The concept generates a joint precedence graph including all variants of a product directly. The graph is automatically derived from the bill of materials and buildability rules as well as existing solutions for the assignment of tasks to workstations. The presented method is very error prone and can improve the practical applicability of many assembly line balancing problems, that could not be used in practice yet.

Producing companies have to cope with an increasing product variety and frequently changing demands. Despite this development assembly lines are used to gain an efficient production process. The main focus in setting up and reconfigurating an assembly line is the assignment of tasks to workplaces. Sophisticated algorithms have been developed for this problem [1, 2].

Almost all existing concepts assume that a precedence graph, describing the relations which tasks have to be done before others, exists. A manual definition of a precedence graph for products with high variety fails, due to the complexity. Some authors developed intelligent methods for an automatic assembly sequence generation [3, 5–11]. Most methods proposed have in common that they start bottom up. Initially no relation exists and all tasks are independent. Restrictions between tasks are added successively. This proceeding is very error-prone as forgetting a relation can lead to an unfeasible setup solution, when the graph is used as an input in an assembly line balancing algorithm. Furthermore, none of the former approaches can handle a product portfolio without an explicit variant definition like it is predominant e.g. in the automobile industry. We propose a method that starts top-down with the existing feasible assignment and sequencing solution and uses automatic algorithms to break up restrictions successively. The algorithms use data that already exists in many producing companies. The first advantage of the top-down approach is the guarantee that in any stage of the precedence graph development, valid assignment solutions can be derived from the graph. There are no possibilities to change the order of tasks for assignment algorithms in the graph that could lead to unfeasible

assignment solutions. Second, the approach can deal with a practical number of variants as the joint-precedence graph, consisting of all models produced in an assembly line, is created directly. Third, the method uses the existing product documentation.

1 Problem Statement

A precedence graph $G = (N, E, t)$ where N is the set of nodes, E the set of edges and t a weightingvector, can be used for different purposes. Within assembly line balancing problems, the precedence graph restricts the possible assignment solutions of tasks to workplaces. A mathematical formulation of a restriction in such a combinatorial problem, assuming an ascending numbering of the workstations, is given in equation 1, where a task a with a precedence relation to task b has to be assigned to an earlier workstation in the assembly line than task b. The set ST includes all workstations st and the binary variable $x_{a,st}$ describes the assignment of task $a \in N$ to station $st \in ST$.

$$\sum_{st=1}^{ST} x_{a,st} \cdot st \leq \sum_{st=1}^{ST} x_{b,st} \cdot st \quad \forall \quad a, b \in N \quad | \quad e_{a,b} \in E \tag{1}$$

In the considered mixed model-case where different variants are produced in one assembly line, the precedence graph has to include the relations of all tasks from all products. The usual way to do this is generating a precedence graph for each of the variants and joining them ([12, p.228ff]). In a scenario with high product variety, e.g. in the automobile industry with a theoretical maximum of 10^{32} different variants [13, p.447], an approach is needed that generates the joint precedence graph directly.

1.1 Data-Basis

The concept is based on three types of input that are used by different algorithms. Even if one type of input is missing in an application of the concept, the other methods can still be processed as they are all independent from each other.

The **open variant bill of materials** fulfills the requirements for a product with high variety and thus is used in many manufacturing companies [14, p.266]. A typical implementation of this product documentation is done by so called codes and code-rules [15, p.1012ff]. Codes represent the product features a client can order, e.g. R1 for a standard radio and R2 for a Radio with GPS. A specific variant is described by a so called code bar, consisting of a combination of available codes. For the identification which parts are to be assembled in a certain variant, each part has its own code-rule assigned, that describes in which case it is needed. These code-rules are boolean expressions of propositional logic. An examplary code-rule which describes for a part that it is needed if engine M1 or M2 and the radio R2 but not the air conditioning K1 is selected could look like this: $(M1 \vee M2) \wedge R2 \wedge \neg K1$. The tasks in the assembly line are connected

with the different parts in the BOM that are needed for an execution of the tasks. Thereby it is possible to identify the necessary tasks to produce a certain variant. Further details can be found in [16]. Based on the code-rules the product structure also includes information about dependencies between parts/processes that exclude each other in order to represent a specific product feature. In general any other product documentation type that fulfills this requirement can be used for the methods to be presented later on.

The **buildability rules** set defines which parts or part sets are compatible. Many producing companies use these rules for a reduction of the set of variants a client can order. For example it could be defined that the Radio with the GPS system is not offered with the smallest engine (i.e.: $M1 \rightarrow -R2$).

This concept is also used to model technically unavailable options as well as simple exclusive options like the engines, which can exist only once in each car.

The **assignment of tasks to different zones of the product** which could be the front, the back, the sides and the inside of e.g. a car or any other product is another input for the following methods. In the previously described open variant BOM, this assignment is already existing in the product documentation. The different zones should be selected in a way that it can be assured that tasks from different zones do not depend on each other and can be processed independently.

Over time **different assignment solutions** are generated, as a regular reassignment of tasks to workplaces is necessary [17, 23]. This is especially true if the product is available in many different variants and the demand for certain options changes over time. Apart from this assignment information, a sequencing solution of the tasks on every workplace is required.

2 Literature Review

The oldest way of building a precedence graph is to create it manually, relation by relation [18, p.243], [19, 4]. This is done by asking the question:"Is there a precedence relation between process a and process b?". As the number of questions would be too high for a product with many tasks, [3] and [4] invented methods with more complicated but less questions. The drawback is that the questions are too complicated to answer even with products of low complexity, which leads to many errors [4], [5, p.310]. Many approaches ([7][8]) use geometrical data or CAD-Data ([5][6]) to do an automatic disassembly of a product in all possible sequences. Other concepts use self-defined data-structures which are only a preliminary step to get a precedence graph ([3, 9, 10].

To sum up, many remarkable efforts have been made to generate precedence graphs. But, in practice, many of them generate problems. A manual definition of all relations for a product with thousands of parts is impossible and error-prone. CAD-Data based approaches need precise data. Approaches that need additional data-structures would even increase the complexity of the problem, as another data-basis must be maintained.

3 Concept

The main difference to existing approaches is that a top-down approach to the problem is suggested. Instead of starting with all tasks being independent from each other, we begin with an existing assignment sequencing solution, that gives us an initial joint precedence graph representing the whole product portfolio. This graph is generated by creating the precedence relations from each task i to its successor $i+1$. Provided that the sequence of the workstations is known, the order of the tasks can be derived from the existing sequencing solutions on the workplaces. This initial graph is basically a linear list, without any degree of freedom. If it was used in an assembly line balancing process, only this assignment and sequencing solution would be feasible and the balancing method would be unable to change the given assignment. Starting from this situation three steps are processed to eliminate restrictions to identify other valid sequences of tasks:

1. Splitting the set of tasks into independent subsets
2. Identifying mutually exclusive tasks
3. Merging sequencing solutions from the past

3.1 Splitting into independent subgraphs

First the tasks are classified into different independent working-zones to build parallel subgraphs. If the open variant bill of materials is used, as described in 1.1, parts and thereby processes are already assigned to certain positions on the workpiece. Therefore, only a relation between positions and independent working-zones has to be defined. Introduced AND nodes are used to represent the parallelism of two subgraphs. This means that all branches in the AND-construct are independent and can be parallelized. After that, four steps to generate the constructs are undertaken. The main steps of the procedure are illustrated in figure 1.

1. Add all direct precedence relations between tasks belonging to the same working zone. These can be derived from the transitive paths.
2. Delete all precedence relations between tasks that belong to different independent working-zones.
3. Add AND-Open as new start node and AND-Close as new end node
4. Add precedence relations between the AND-Open and all first nodes in the different branches as well as relations between all last nodes in the branches to the AND-Close node.

3.2 Identify mutually exclusive tasks

The following algorithm makes use of the fact that no precedence relations can exist between mutually exclusive tasks. E.g. only one engine can be built into a car. Accordingly, all precedence relations between two tasks that are necessary for two different types of engines can be eliminated.

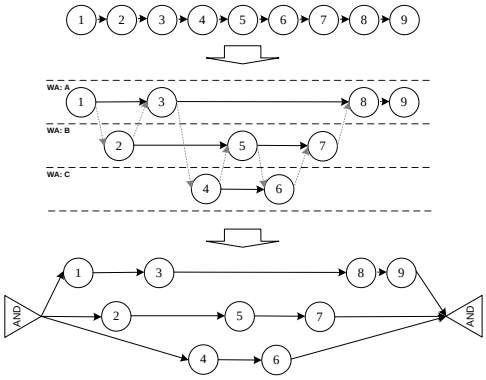

Fig. 1. Splitting of tasks into independent subgraphs

There are two options to identify mutually exclusive tasks. First, the code-rules of the parts, connected to processes, can be checked. Second buildability rules that define, which codes cannot be built together into one product can be analyzed.

The **code-rules** are transfered into the disjunctive normal form. Each conjunction in the first code-rule is checked against all conjunctions of the second code-rule. If a negation of a code c_i in the code-rule CR_i of part i is found in one of the codes c_j in the code-rule CR_j in part j, mutually exclusiveness between the parts i and j can be stated.

$$ME(i,j) = \begin{cases} 1 & if \quad c_i \neq \neg c_j \quad \forall \quad c_i \in CR_i, c_j \in CR_j \\ 0 & else \end{cases} \tag{2}$$

The **buildability rules** set defines directly which codes are mutually exclusive. Accordingly, two processes which need mutually exclusive parts are incompatible, cannot occur in one variant and all precedence relations between them can be eliminated.

As most processes are not dedicated to handle only one single part but sets of parts, the check for mutual exclusion of different processes has to be done matching all items of the sets of parts connected with the processes. In order to model the exclusion in the graph an XOR-construct with the nodes XOR-Open and XOR-Closed is introduced. Only one branch in these constructs can be needed for a single product. This means that the branches are mutually exclusive (see figure 2).

3.3 Merge sequencing solutions from the past

As in modern assembly lines many different variants are produced and demand for variants changes over time, a regular reconfiguration of the assembly line and therefore reassignment of tasks is necessary. All of these assignment and

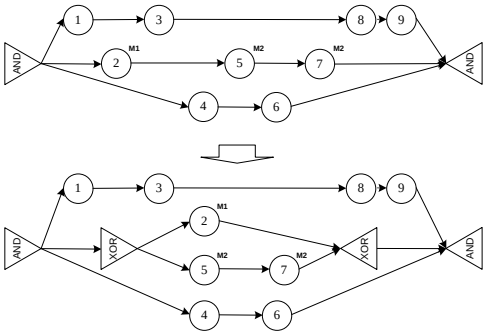

Fig. 2. Identifying mutually exclusive tasks

sequencing solutions are valid in terms of their precedence relations. Otherwise they could not have been implemented in the assembly line. These additional solutions can be used to identify further degrees of freedom in the precedence graph. For this purpose, the methods previously described are executed for the new assignment and sequencing solution that is to be analyzed. First, the resulting graphs are being united. Formally two graphs are united by unifying their nodes and their edges: $G_i \cup G_j = (N_i \cup N_j, E_i \cup E_j)$.

By this, cycles can come up, if $(n1, n2) \in E_i$ and $(n2, n1) \in E_j$. That means that both precedence relations can be omitted as both orders are possible for the involved tasks. Accordingly, cycles indicate degrees of freedom in united precedence graphs. These cycles can include more than two tasks and can even overlap. Therefore a matrix is built with an algorithm implementing a depth-first search through the precedence graph. The entries $p_{i,j}$ describe if a path between the two nodes i and j exists. A resulting relation in both directions means, that they are invalid and can be deleted.

$$p_{i,j} = \begin{cases} 0 & if \quad \exists \text{ path between } i \text{ and } j \\ 1 & else \end{cases} \tag{3}$$

An AND-construct is used, as shown in figure 3, for the separation of the processes.

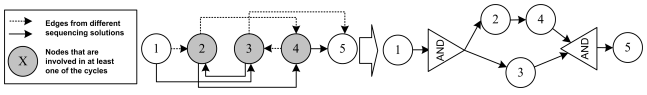

Fig. 3. Dissolving cycles

This step is repeated with all assignment solutions available and at any time a new valid sequencing solution is found. Successively the precedence graph is improved like a knowledge base by the new sequencing and assignment solutions

that are generated regularly in the reconfiguration process. Another benefit is that new tasks, just added to the assembly line are correctly integrated into the graph.

4 Results and Conclusion

The described concept was tested in a real-life scenario at a car manufacturer. An assignment and sequencing solution for an assembly line with 30 workstations and 578 tasks as well as four different assignment solutions from the past were used to generate a precedence graph automatically. For an additional manual editing process of the graph it is important that the complexity and therefore the number of tasks to be considered concurrently, is reduced. As the AND-branches, generated by step 1 of the algorithms, are independent from each other, the number of tasks to be looked at equals the number of tasks in the branches. The branch with the highest number of tasks contained only 28,4% of the total number of tasks. The nodes in one AND-construct, built by the splitting of tasks into different independent working-zones, are structured further by the second step, inducing an identification of mutually exclusive tasks. The resulting XOR-constructs reduce the number of tasks, that have to be kept in mind, even further. In the example the amount of tasks to be looked at per AND-branch were reduced additionally by 11% in average. In the example, the use of automatic assembly line balancing algorithms reduced the number of floater deployments necessary to produce the given car sequence by 10,3% in comparison with the manual assembly line configuration. Floaters are skilled workers which are reserved to support production if workstations reach their workload limits. The number of concurrent floater deployments, defining the necessary number of floaters to be hold ready, was reduced by 16,6%. It is thereby shown that enough degrees of freedom were identified in the graph for a successful application of automatic algorithms for a reconfiguration process.

To conclude, the presented method generates a precedence graph automatically by analyzing existing product documentation. It can be guaranteed that the precedence graph includes only really existing degrees of freedom. This assures that an assembly line balancing algorithm can generate only feasible solutions. It was shown that enough degrees of freedom were extracted to improve line balance. Still, precedence relations do exist in the graph that are not necessary, which restricts the solution space. In our further research we concentrate on a top-down approach for a manual analysis that eliminates more precedence relations and discovers even more degrees of freedom.

References

1. N. Boysen, M. Fliedner, A. Scholl. Sequencing mixed-model assembly lines to minimize part inventory cost.OR Spectrum, 192:349–373 (2009)
2. A. Scholl, C. Becker. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. Eur. Jour. of Operational Research,168:666–693 (2006)

3. A. Bourjault. Contribution a une Approch Methodology de L'assemblage Automatise: Elaboration Automatique des Sequences Operatoires. Universite de Franche-Comte (1984)
4. T.L. De Fazio, D.E. Whitney. Simplified generation of all mechanical assembly sequences. IEEE Journal of Robotics and Automation, RA-3(6):640–658 (1987)
5. R.H. Wilson. Minimizing user queries in interactive assembly planning. IEEE Transactions on Robotics and Automation, 11:308–312 (1995)
6. R.l E. Jones, R. H. Wilson, T. L. Calton. Constraintbased interactive assembly planning. Proc.: IEEE Int. Conf. on Robotics and Automation, 1:913–920 (1997)
7. S. G. Kaufman, R. H. Wilson, R. E. Jones, T. L.Calton. The archimedes 2 mechanical assembly planning system. Proc.: IEEE Int. Conf. on Robotics and Automation, 1:3361–3368 (1996)
8. A. C. Sanderson, L. S. Homem de Mello, and H. Zhang.Assembly sequence planning. AI Magazine, 11:62–81 (1990)
9. M. Santochi, G. Dini. Computer-aided planning of assembly operations: the selection of assembly sequences. Robotics and Computer-Integrated Manufacturing, 9:439–446 (1992)
10. Y. Yokota, D.R. Rough. Assembly/disassembly sequence planning. Assembly Automation, 12:31–38 (1992)
11. Y. Cho, C.K. Shin, and H.S. Cho. Automated inference on stable robotics assembly sequences based upon the evaluation of base assembly motion instability. Robotica, 11:351–362 (1993)
12. W. Domschke, A. Scholl, S. Vo. Produktionsplanung.Springer Verlag (1993)
13. H. Meyr. Supply chain planning in the german automotive industry. OR Spectrum, 26:447470 (2004)
14. W. Dangelmaier. Produktion und Information-System und Modell. Springer (2003)
15. A. Roeder. A methodology for modeling inter-company supply chains and for evaluating a method of integrated product and process documentation. Eur. Jour. of Operational Research, 169:1010–1029 (2006)
16. C. Sinz. Verifikation regelbasierter Konfigurationssysteme. Fak. fuer Informations- und Kognitionswissenschaften, Eberhard-Karls-Univ. Tuebingen (2003)
17. N. Boysen, M. Fliedner, A. Scholl. Production planning of mixed-model assembly lines: Overview and extensions. Tech. rep., Friedrich-Schiller-University Jena (2007)
18. C.L. Chen. Automatic assembly sequences generation by pattern-matching. Technical report, School of Engineering and Technology, Electrical Engineering and CAD/-CAM Center, Purdue University, 1989.
19. W. Jentsch, F. Kaden. Automatic generation of assembly sequences. Artificial Intelligence and Information-Control Systems of Robots, 1:197–200 (1984)
20. D.F. Baldwin, T.E. Abell, M.-C. Lui, T.L. De Fazio, and D.E. Whitney. An integrated computer aid for generating and evaluating assembly sequences for mechanical products. IEEE Trans. Robot. and Automat., 7:78–94 (1991)
21. J.M. Henrioud and A. Bourjault. Computer-Aided Mechanical Assembly Planning, chapter LEGA - A computer-aided generator of assembly plans, pages 191-215. Kluwer Academic Publishers (1991)
22. G. Dini, M. Santochi. Automated sequencing and subassembly detection in assembly planning. Annals CIRP, 41:1–4 (1992)
23. E. Falkenauer. Line balancing in the real world. In Int. Conf. on Product Lifecycle Management (2005)