Aggregation – the Key to Integrating Production Planning and Scheduling

J. Váncza[1] (2), T. Kis[1], A. Kovács[2]
[1]Computer and Automation Research Institute, Hungarian Academy of Sciences
[2]Budapest University of Technology and Economics, Budapest, Hungary

Abstract
In this paper we suggest an integrated planning and scheduling framework with a special emphasis on the link between these control levels. Our planning model is generated automatically by performing aggregation on *de facto* standard product and technology related data in the dimensions of time, resource capacities and operations. The method addresses make-to-order production environments. An industrial case study is also presented, demonstrating how our algorithms work on large-scale problem instances.

Keywords:
Production, Planning, Scheduling

1 INTRODUCTION

Production planning and scheduling (PPS) map the load of a factory to its capacities on different time horizons and levels of detail. Planning and scheduling problems differ in their timescale, the granularity of resource and activity models and in their optimization criteria.

The two levels of PPS are strongly coupled since planning sets the goals as well as the resource and temporal constraints for scheduling. On the other hand, scheduling is responsible for unfolding a plan into detailed resource assignments and operation sequences. No scheduling strategy can improve much on an inadequate plan, whereas a bad scheduling strategy that wastes resources may inhibit the fulfillment of a good plan. All this makes PPS extremely complex and hard to solve. At the same time, PPS calls for efficient decision support methods and intuitive, flexible models with fast, reliable solution techniques that scale-up well to large problem instances.

Departing from detailed product, resource and production technology related information, **aggregation** connects the two levels by composing distinct resources and operations into larger units. Thereby aggregation reduces the complexity of production planning when deciding over the flow of materials and the use of resources on a longer horizon. Aggregation addresses also the various uncertainties of production: it does not allow generating detailed plans for a future that will certainly be different from what we anticipate now. Note that this principle applies well not only in PPS but also in the early stage of product design [1] and engineering [2].

The idea of aggregate production planning was introduced almost fifty years ago, just with the motivation to respond to fluctuations in product orders by means of a clear-cut mathematical model that used a common measure of work required by the different orders [3]. A number of theoretical models followed: they merged discrete operations requiring the same resources into distinct aggregate activities [4]. However, if the parts loop over the same resources several times, this method may result in very complex temporal interdependencies of the activities. Consequently, temporal patterns used by planning could hardly be filled in with technological data [5] [6], and planning disregarded most temporal relations [4]. In practice, so-called material/manufacturing requirements planning systems (MRP/MRP II) do work with precedence relations, but assume that components and complete products can be produced with fixed lead times, without any direct regard to capacities and the actual load. None of these assumptions is realistic in make-to-order production environments that respond to fluctuating orders. No wonder that plans generated this way can barely be refined to executable detailed schedules.

2 PROBLEM STATEMENT

In this paper, we consider aggregation as the process that provides a mapping between the models of PPS problems. Hence, it is the solution of a **representation problem** that has an essential impact both on planning and on scheduling. The main requirements towards aggregation are as follows:

- Planning must respect the main temporal constraints of orders (e.g., due dates) and the resource capacity constraints of the factory.
- Production plans should be unfoldable into executable schedules. Planning must also handle precedence relations that ensue from complex product structures (e.g., assemblies) and technological routings.
- However, resource assignment problems with finite capacities and precedence constraints are in general extremely hard to solve. Due to aggregation, typical instances of planning problems should be of the size and complexity that can be solved efficiently.
- Planning and scheduling models should be built by using common product and production data (e.g., bills of materials (BOMs), routings, resource calendars). Open orders should be addressed at both levels.
- Albeit aggregation removes details – due to the differences between the planning and scheduling model – it may introduce new constraints that distort the original problem. The effect of such constraints should be kept as small as possible.

The clarification of these interdependencies makes PPS methods more transparent and efficient. Hence, advanced PPS methods could better be applied in supply chain logistics, too [7]. We emphasize that the actual PPS frameworks used in factories should meet the above general requirements.

In what follows, we introduce shortly our PPS approach (Sect. 3), analyze the role of aggregation, define an optimal activity model, and provide algorithms to construct such activities in Sect. 4. Further, we present an industrial case study, whose lessons are summed up in Sect. 6.

3 INTEGRATED PLANNING AND SCHEDULING

3.1 Project-based production planning

Recently we have suggested a novel approach to modeling and solving production planning problems in make-to-order production environments [8]. This method – as other project-based planners [5] [6] – unifies the capacity and the material flow oriented aspects of planning. Orders are modeled as **projects** that compete for limited resources. Projects have time windows set by their release dates and deadlines. Each project is represented as a network of **activities** that are linked by various precedence constraints. An activity may require several **resources** and the execution of a given amount of work. However, the intensity of executing an activity may vary over time; the activity can even be pre-empted. Activities are **aggregates**: they represent groups of manufacturing, assembly, etc. operations, some of which are executed simultaneously, some sequentially, and others independently of each other. This leads to a model in which not the lead times but only the work amounts of activities are fixed a priori.

The required resources are typically both machine and human resources that should be shared by activities of different projects. The resources may be distributed, geographically dispersed and may even belong to different organizations. The **capacities** of resources are limited and may vary over time.

Plans are generated on a medium term horizon with one week's time unit. The solution, i.e. a production plan specifies what portions of the activities have to be performed and how much external resources must be used in each time unit so that all temporal, precedence and capacity constraints are respected. This solution can be optimized according to various cost and due-date performance criteria. In the running example, the primary objective is to minimize the cost of external resource usage, while the secondary objective is to keep work-in-process (WIP) level as low as possible.

Our production planner works with customized, powerful mathematical programming methods. The solver developed specifically for the project model efficiently solves problems of real-life sizes [8].

3.2 Constraint-based job shop scheduling

Our short-term scheduler performs finite capacity scheduling with respect to detailed technological constraints. The scheduling horizon is as long as the time unit of the planner (i.e., one week), while the scheduling time unit is 0.1 hour. The operation sets to be scheduled are given by the aggregate activities that fall into a given time unit in the medium-term production plan. Typically, schedules are generated for the next few weeks only. If an activity covers several weeks, then its operations are distributed in this period proportional to the activity's intensities.

There are both individual (e.g., machine tools) and group resources (homogeneous machine groups, assembly stations, various pools of qualified workforce). Resource availability – that may vary shift-by-shift – is given by a calendar. Resource and time requirements, as well as the sequence of operations are described in the routings. Each operation requires a given combination of resources. E.g., a turning operation might require a turning centre and a machinist during the entire length of its processing. Operations have specific processing, setup and transportation times.

The solution is an assignment of starting times to operations such that all temporal, precedence and resource constraints are satisfied. The main objective is to minimize the maximal tardiness with respect to the due dates set by planning. We obtain such a solution using a **constraint-based scheduling** approach [9].

4 AGGREGATION OF PROJECT TREES

4.1 The project tree

Each project can be described by a rooted tree, the so called **project tree** (denoted by T), whose vertices represent manufacturing operations (see Fig. 1). Vertices with several children denote assembly operations, while those with a single child represent either machining operations or joining a purchased part to the workpiece. The execution of the project over time advances from the leaves towards the root that stands for the finishing operation of the final product. Edges represent strict precedence relations, i.e., the sons of an operation must all be completed before the operation itself could be started.

Each operation i of the project tree has a processing time p_i , setup time s_i and transportation time u_i . Resources needed by i are given by $R_i \subseteq R$ where R is the set of all available resources. We assume that transportation and setup is performed before the operation, but while the first needs the part only, setup requires solely the resources of the operation.

Fig. 1 below shows the parameters and structure of a sample project. The root operation #1 is a manual assembly that produces the final product. The project tree contains also other assembly operations (#3 and #8) while the other nodes correspond to various machining operations. For instance, operations #1 and #3 need the same human and machine resources, actually a welding cabinet with appropriate personnel.

i	p_i	s_i	u_i	R_i
1	12.0	3.0	0.5	M1, H1
2	6.0	1.0	0.5	M2, H2
3	9.5	3.5	0.5	M1, H1
4	8.0	1.0	0.2	M3, H2
5	6.0	1.0	0.2	M2, H2
6	9.0	1.0	0.2	M3, H2
7	2.5	1.0	0.5	M2, H2
8	5.0	2.0	0.5	M4, H1
9	9.0	1.0	0.5	M2, H2

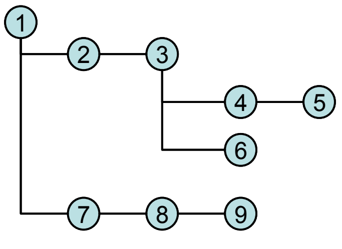
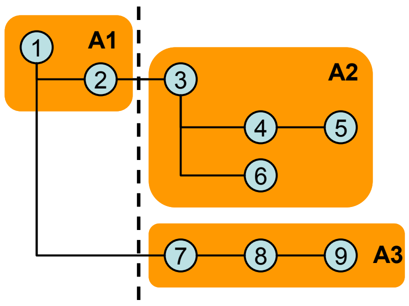


Figure 1: A sample project.

4.2 Activity models of the project

At aggregation, connected vertices of the project tree are contracted into components that define the activities of the planning model. This partitioning of the project tree is called the **activity model** of the project. If two operations of the project tree that are connected by a precedence constraint are inserted into the same activity, then this constraint is omitted from the aggregate model. Otherwise, a precedence constraint is posted between the two aggregate activities. Note that the precedence graph of the activities will also form a tree. The resource requirements of an activity are the sums of the p_i and s_i times of the contained operations per each resource required. Fig. 2 shows a possible activity model of our sample project.



	M1	M2	M3	M4	H1	H2
A1	15.0	7.0	-	-	15.0	7.0
A2	13.0	7.0	19.0	-	13.0	26.0
A3	-	13.5	-	7.0	7.0	13.5

Figure 2: An activity model of the sample project and the resource requirements of the activities.

Clearly, this method guarantees that aggregate activities can be unfolded to feasible discrete operation sequences. Different partitionings of the project tree result in various activity models; to characterize and find the best one we introduce now some new properties.

Each activity A of the activity model P has a weight $W(A)$ that is an estimate of the throughput time needed for executing all the operations of A. Cardinality of the activity model is denoted by $q(P)$. Height $h(P)$ of the tree P is the length of its longest leaf-to-root paths. Finally, $\Pi(A)$ denotes the set of leaf-to-root directed paths in the project sub-tree corresponding to activity A.

4.3 The effects of aggregation on planning and scheduling

Merging operations of the project tree into larger activities decreases the computational complexity of the planning problem. On contrary, too large activities can hardly be unfolded into feasible short-term schedules. Therefore, it is reasonable to set a limit to the weight of the activities. It can be proven that the best compromise of these conflicting criteria is setting the activity weight limit to the length of the aggregate time unit. If an operation with a longer processing time hurts this condition, then this operation constitutes a single activity.

Though the planning problem is usually considered as a relaxation of the job-shop level problem, some extra constraints may be introduced or strengthened due to the aggregation step. In any case, a precedence constraint states that the connected operations or activities have to be executed in the given order, in **distinct time units**. Hence, a precedence constraint implies a time unit change between finishing the preceded and starting the preceding activity. Therefore, the lead time of a project using the activity model P cannot be less than its height $h(P)$. Consider the alternative activity models of the same project at Fig. 3: $q(P_1) = q(P_2) = 5$, but their heights are different: $h(P_1) = 4$, while $h(P_2) = 2$. Hence, P_2 provides a more appropriate aggregation of the same project tree.

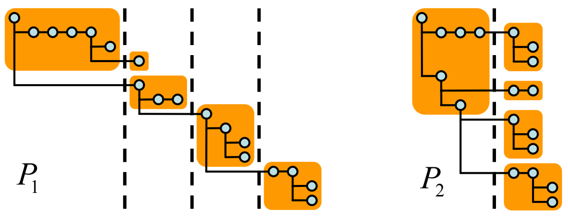


Figure 3: Alternative activity models.

Obviously, in order to respect the deadlines and to keep WIP low, we are interested in finding the activity model P with minimal height $h(P)$. Note that this means increasing the parallelism between activities.

With the above considerations in mind, the definition of the **optimal activity model** of a project is as follows:

- An activity A is a connected component of the project tree such that its estimated throughput time expressed by $W(A)$ fits in one aggregate time unit.
- The optimal activity model P of a project tree T is a partitioning of the project tree into activities such that $h(P)$ is minimal. With the above prerequisite, $q(P)$ is also minimal.

4.4 Generating the optimal activity model

At the time of activity formation, it is hard to compute the throughput time of an activity. On the one hand, the set of activities present at the factory concurrently with the one on issue is not known at this phase. On the other hand, determining the minimal throughput time of a single activity is an NP-hard resource-constrained project scheduling problem [10]. Hence, we suggest heuristic weight functions for estimating the throughput time of an activity. In order to allow for the resource requirements of the other activities and the various uncertainties of production at the time of aggregation, these weight functions are increased by a **security factor** $sf \geq 1$.

$$W_1(A) = \sum_{i \in A} (p_i + \max(u_i, s_i)) \times sf\ ,$$

$$W_\infty(A) = \max_{\pi \in \Pi(A)} \left(\sum_{i \in \pi} p_i + u_i \right) \times sf\ .$$

W_1 is based on the assumption the operations of the activity must be processed sequentially. In this case, the transportation and the setup times of each operation can overlap. W_∞ assumes that the factory possesses unlimited capacities, hence the weight of the activity can be estimated with its longest leaf-to-root path.

In production environments where both resource constraints and complex precedence relations should be accounted for, we suggest a third weight function: $W_{sch}(A)$ estimates the throughput time of operations $i \in A$ by the application of a priority rule based scheduler. We use the “greatest rank positional weight*” (GRPW*) rule [10]. It handles multiple resource requirements and can be extended to account for transportation and setup times belonging to operations.

The generation of optimal activity models corresponds to solving a minimal height – minimal cardinality weight-bounded tree partitioning problem. In [11], we suggested polynomial-time bottom-up algorithms for solving such problems. For weight functions W_1 and W_∞, the exact minimization of height and cardinality is possible using a dynamic programming approach. For the weight function W_{sch}, the suggested algorithm for the exact minimization of height calls the $O(n \log n)$ complexity scheduler $O(n)$

times in the worst case, and it can be extended by a greedy algorithm to decrease cardinality.

For the sample project of Fig. 1, using an activity weight limit of 40 hours, with a security factor of 1.25, the bottom-up algorithm first realizes that the sequence of operations #4 and #5 fits into one activity by a heuristic. Checking that this activity can be expanded to contain {#3, #4, #5, #6} requires running the rule-based scheduler, $W_{sch} = 36.25$. However, operation #2 cannot be added to this activity, $W_{sch} = 44.375$. In the same way, the activities {#7, #8, #9} and {#1, #2} are created, leading to a partitioning P with $h(P) = 2$ and $q(P) = 3$ (see Fig. 2). The greedy heuristic could not further decrease the cardinality, since in this case, the previous partitioning was optimal for cardinality, too.

5 INDUSTRIAL APPLICATION

The algorithms presented above constitute a module of an integrated production planner and scheduler system that has already been tested with industrial data.

The factory of our case study manufactures mechanical components of high value in a make-to-order manner, by using machining and welding centers, assembly and inspection stations. It has cc. 100 individual and 50 group resources. A typical project consists of 20 to 500 discrete manufacturing operations, with processing times in the range of 0.5 to 120 hours. Operations require both machine and human resources. The project trees were generated from *de facto* standard BOM and routing databases of the factory. From these trees there were generated optimal activity models consisting of 1 to 10 activities. The horizon of the planning problem was set to 15-25 weeks. Using the activity models, we could create valid production plans and schedules for all the cc. 300 running projects. Applying W_{sch} with $sf = 1.25$, the schedules always had zero tardiness. Fig. 4 presents a fragment of a production plan: white areas show the allowed time windows of the projects and dark fields show the weeks when activities of the projects have to be performed. There is a one-week pause at the factory around the end of the year.

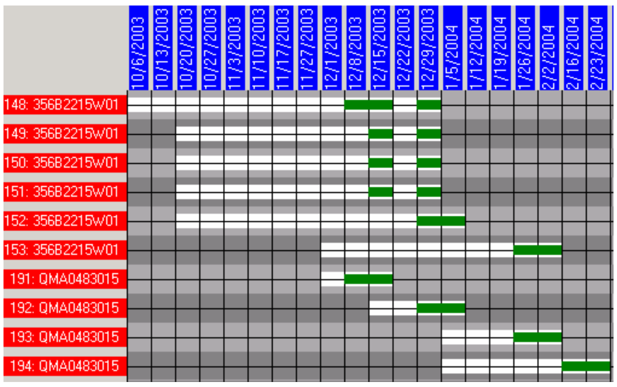


Figure 4: Details of the project view of a production plan.

The main lessons of our experiments are as follows: Project trees generated with the scheduling weight approximation W_{sch} were the most appropriate because the production plans could be unfolded to feasible schedules in every case. In contrast, W_{∞} typically overloaded the factory: it often turned out that works planned for some week required even 10-20% more time after detailed scheduling. We did not apply W_1 because there were many discrete resources and the project trees had a balanced tree structure, with several parallel branches. Setup and transportation times had an important role; without them the production plans were too optimistic, consequently, the schedules generated were too tight and practically not executable. Now we are just preparing simulation experiments in order to have a better understanding of how slacks can compensate for the effects of unanticipated uncertainties.

6 CONCLUSIONS

In this paper we have emphasized the role of aggregation as the primarily link between the models of production planning and scheduling. We have pointed out that aggregation is a representation problem whose solution has a great impact both on the quality of the production plans and the feasibility of the detailed schedules. The proposed methods enable PPS to work on common product, resource and production technology data. Our experiences support the claim that proper aggregation is a major prerequisite for using advanced PPS methods successfully.

7 ACKNOWLEDGMENTS

This work has been supported by the “Digital Factory” NRDP grant No. 2/040/2001. We are thankful to András Márkus for his comments and Péter Egri for his technical support.

8 REFERENCES

[1] Maropoulos, P.G., McKay, K.R., Bramall, D.G., 2002, Resource-Aware Aggregate Planning for the Distributed Manufacturing Enterprise, Annals of the CIRP, 51/1:363-366.

[2] Mentink, R.J., van Houten, F.J.A.M, Kals, H.J.J, 2003, Dynamic Process Management for Engineering Environments, Annals of the CIRP, 52/1:351-354.

[3] Holt, C., Modiglinai, F., Simon, H.A., 1955, A Linear Decision Rule for Production and Employment Scheduling, Management Science, 2/1:1-30.

[4] Bitran, G.R., Tirupati, D., 1993, Hierarchical Production Planning, In: Graves, S.C., Rinnooy Kan A.H.G., Zipkin, P.H. (eds), Logistics of Production and Inventory, 523-568, Elsevier.

[5] Hackman, S.T., Leachman, R.C., 1989, An Aggregate Model of Project-Oriented Production, IEEE Trans. on Systems, Man and Cybernetics, 19/2:220-231.

[6] Hans, E.W., 2001, Resource Loading by Branch-and-Price Techniques, Ph.D. Thesis, Twente University Press.

[7] Wiendahl, H.-P., von Cieminski, G., Begemann, C., 2003, A Systematic Approach for Ensuring the Logistic Process Reliability of Supply Chains, Annals of the CIRP, 52/1:375-380.

[8] Márkus, A., Váncza, J., Kis, T., Kovács, A., 2003, Project Scheduling Approach to Production Planning, Annals of the CIRP, 52/1:359-362.

[9] Baptiste, Ph., Le Pape, C., Nuijten, W., 2001, Constraint-Based Scheduling, Kluwer.

[10] Demeulemeester, E.L., Herroelen, W.S., 2002, Project Scheduling: A Research Handbook, Kluwer.

[11] Kovács, A., Kis, T., 2004, Partitioning of Trees for Minimizing Height and Cardinality, Information Processing Letters, 89/4:181-185.