Abstract

This page seeks to explore the computability of planning models in continuous production.

Intro

A production planning problem considers processing items on machines in time horizon **T**. An operation plan (or schedule) should be made to satisfy orders in time. This problem is critical in supply chain management, and it has traditionally been divided into multiple parts, including S&OP, AP (aggregate planning), MRP, and scheduling at the lowest level. None of these above can provide a comprehensive solution to production plan, inventory and delivery.

In 2016, I worked with Texas Instrument on a project to reduce WIP levels in their semi-conductor facilities. The precise scope of the project lies between traditional MRP and AP, as it focuses on materials production, and inventory levels. Later in 2019 I worked with a large ICT company on an enormous planning problem, which considers over 100,000 products with complex BOM in a planning horizon over 3 months. The project should provide the company with an integrated plan on inventory, order delivery, production, procurement, out-sourcing, and so on, which makes it encloses S&OP, AP, and even procurement plans.

It seems that supply chain operations are going to discard the methodology defined by traditional management tools. To translate strategic plan into operations, we nowadays prefer to consider what has been treated separately as a whole. It is partly the comes from the development and success of AI, machine learning, and large-scale mathematical programming.

The keyword becomes the ***computability***. In the TI project, most instances define weekly plans, and can be solved within 3,600s by open-source linear optimizer *CLP*. For the project I did in 2019, we use LP models with over 1 billion decision variables, the computation time goes up to 3~4 hours, for a rolling planning model it is merely acceptable. But they are all linear optimization models. Using LP means you cannot have integer variable, disjunctive constraints, SOS, and so on, which prevents you from modeling operation rules **precisely**. At the same time, fractional solutions are not acceptable for many cases, and you will need a good rounding heuristic to play with the integer issues. Generally, extra efforts should be placed on LP-specific constraints, systematically tuned hyper-parameters, and the designed solving procedure since you cannot model prioritized "multi-objective" requirements in LP.

Compared to continuous production, discrete manufacturers would like to pay more attention on using models to solve production planning problems, especially for industries like semi-conductor, ICT, mobile phones and so on. The reasons could still be the problem of computability. For discrete production, one usually wants to minimize inventory levels to achieve Just-in-Time style solutions. The ideal way is to model the production problem by using time-indexed variables; the description of constraints seems to be quite straightforward. It looks like lot-sizing, assignment, and network flows, and can be solved by LP; at least a fractional amount of production makes some sense. You can find detailed reference from Wolsey's planning book[1]. Furthermore, a time-indexed model tends to produce very discrete solutions. A production plan would not be continuous, which is not acceptable in continuous production. The continuous companies would like to minimize setup costs and to keep running the same production, and this goal somehow contradicts the nature of time-indexed models. In this case many rules and descriptions of the problem require MILP-like constraints like routing of jobs, processing time of items and so on. This makes the problem similar to JSP and thus NP-hard.

The Discrete Model

Generally, the goal of a planning systems is to find a solution to plan for production, and ultimately to reduce delivery insufficiency. It can be summarized from high to low by priority: a. minimize production shortage, b. minimize inventory, c. minimize production cost, setup cost, etc.

Notation

We use the following notations.

- set of items: $\mathbf{I} = \{1, 2, ..., N\}$
- set of machines: $\mathbf{L} = \{1, ..., l, ..., L\}$
- demand/order: d_{it}
- lead time: l_i
- suppose there exists resource restrictions on machine r_m
- parents: J_i, a group of items use i as components with rate u_{ij}

Decision:

- production: x_{it} / x_{ilt} - two or three dimensional decision variable
- delivery: s_{it}

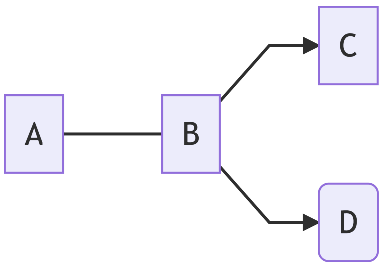
Flow

A MILP/LP model can be defined as follows: t' defines the start time of production that finishes at t, this can be achieved by finding the right mapping from t' to t, for example, the $2-D$ case:

[1] [2]

$$D_{i,t} + M_{i,t} = d_{it} + M_{i,t-1}, t \geq 1$$
$$N_{i,t} + D_{i,t} - \sum_{j \in J_i} u_{ij} X_{j,t} = N_{i,t-1} + x_{i,t'}, t' \leq t$$

This above constraint is usually referred to as ***flow***, i.e., how items are produced and the inventory **flows** through the whole system. For discrete system, the flow actually follows the BOM[2], which are generally a group of trees, here is an example:



The first set of equations keeps track of postponed order by variable M. The second set of equation records the production. For a fractional lead-time that is not able to be rounded, it can be modeled by the method mentioned in my master's report, see [3].

For continuous production, an item is built from a set of operations; for each item i, the building procedure can be represented as a directed acyclic graph $G_i = \mathscr{G}(O_i, E_i)$, where $O_i \subsetneq O$ is the needed operations each of which has to be processed on one of the machines, E_i is the set of directed edges defining precedences. The lead-time here is defined from the processing time by matrix $P_i = \{p_{om}\}$, where p_{om} is the time to finish o on machine m. We will talk about this in the next section.

Resource

The following part considers "coupling" constraints. We have the following possible situations that put rules on a group of items:

- The capacity, fractional type of resources

The kind of resources consumed by machine or group of machines. For example, suppose producing i consumes capacity of machine l by r_{il}, hence:

$$\sum_i r_i \cdot x_{i,l,t} \leq C_{lt}, \forall l, t$$

[2]bill-of-material

- The mold, integral type of constraints

The mold is one kind of resources that are represented as integers, or even binaries. Suppose item i needs a such mold m to be put on machines, whereas the total num of molds are limited, each mold has a capacity q_m, by auxillary variable y representing the number of mold used on line l we have:

$$\sum_{i \in m_i} x_{ilt} \le q_m \cdot y_{m,l,t}, y \in \mathbb{N}^{M \times |L| \times |T|}$$

$$\sum_{l} y_{m,l,t} \le N_m, m \in M$$

Since you have integral variables, the problem is not easy, while it is not as hard as expected.

Shape

Except for flow and resource constraints, there are some rules that requires the ***shape*** of production.

Table 1: Example production sheet, continuous

item	2020-02-01	2020-02-02	2020-02-03	2020-02-04	2020-02-05	2020-02-06	2020-02-07	2020-02-08
0432094			13	13	13			
0433094			17	17	17			
04330AA			227	227	227	448	448	817
043409P						220	220	441
090094	20	20						
0A0108Y	98	98						
0A0109E								51
0A400A8			173	173	173	173	173	206
0A410A8	1352	1352						
0AM30A8			3	3	3	3	3	5
0AQ00AE			42	42	42	42	42	44
0AQ10AE			272	272	272	272	272	272

item	2020-02-01	2020-02-02	2020-02-03	2020-02-04	2020-02-05	2020-02-06	2020-02-07	2020-02-08
05109H						43	347	695
C10F1072	778				444			
C10Q009								635
C116008N	44	44	44	44	44			
C116009								200
C11600A0	90	90	90	90	90			

In real production, it is usually preferable for a line/machine to always work on same items. We call this kind of requirements "**continuity**". In the above production sheet, for example, a continuous production style will result in a constant, steady, production values.[3]

There are many ways to do this, one of which is to use the $\mathscr{L}_1$ method with auxillary variable x^d.

$$x^d_{it} \geq |x_{i,t} - x_{i,t-1}|$$

Another kind of requirements on shape is ***aggregation***. A manufacturer tends to produce as much as possible in one time period instead of producing a series of small batches. This requirement can be solved by tuning the hyper-parameters, including:

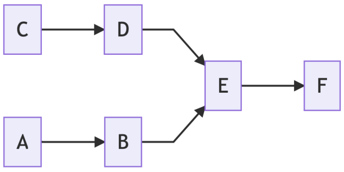
- Use a set-up cost, but this makes the problem hard.
- Distinguish production cost along the time horizon. Let the cost be smaller at certain periods.

The tuning procedure can be done by using simple searching, sorting procedures, when the rules goes crazy, you may need a separate **optimization** model to find feasible objective parameters.

The Continuous Model

While the terms like machines, lines, molds sound "specifically" for discrete production, continuous production (***CP***) is more *operation*-oriented. Just like we mentioned above, the flow of a CP system is generally a DAG.[4]

[3] 0AQ10AE is considered to be continuous.
[4] It is possible to have loops in the operation graph, we do not talk about this here.



There are several ways to do this:

- JSP style: using binary sequence variables to model precedences, see a review paper on JSP formulations, [4]. Unfortunately, it's not easy to extend a disjunctive model so as to adapt for resource constraint.
- discrete style: extend the discrete, time indexed model.

We will talk about discrete style here.

Flow

The theory here is to make a discrete model continuous. The easiest method is to model the item at every single operation as a new item, i.e., if the above graph is for item i, a group of virtual items could be: $[i_C, i_A, ..., i_F]$, where $i_F \equiv i$.

Hence the lead time of i_C, for example equals to the operations time $p(i, C)$, and then you follows the same method in dicrete model. Generally, the shape constraint is *continuous*

From planning to project scheduling

There are some systems where flow cannot apply.

There is no real production **quantity**, more like a project, *planning* is somewhat a *scheduling* problem. This is very common in military situation, like producing an airplane.

We talk about this in later post on project scheduling.

References

[1] M. Pinedo, *Scheduling*, vol. 5. Springer, 2012.

[2] Y. Pochet and L. A. Wolsey, *Production planning by mixed integer programming*. Springer Science & Business Media, 2006.

[3] C. Zhang, J. F. Bard, and R. Chacon, “Controlling work in process during semiconductor assembly and test operations,” *International Journal of Production Research*, vol. 55, no. 24, pp. 7251–7275, 2017.

[4] W.-Y. Ku and J. C. Beck, “Mixed integer programming models for job shop scheduling: A computational analysis,” *Computers & Operations Research*, vol. 73, pp. 165–173, 2016.