3. The Accuracy Tradeoff in Planning

Product flow planning inherently involves some type of mathematical model of operations. That model may as simple as a spreadsheet that computes planned inventory balances, or as sophisticated as a system of multiple optimization models. By the way, a semantic distinction: the model is our abstract numerical representation; the planning tool is the software (or piece of paper in the very simplest situations) that holds our model.

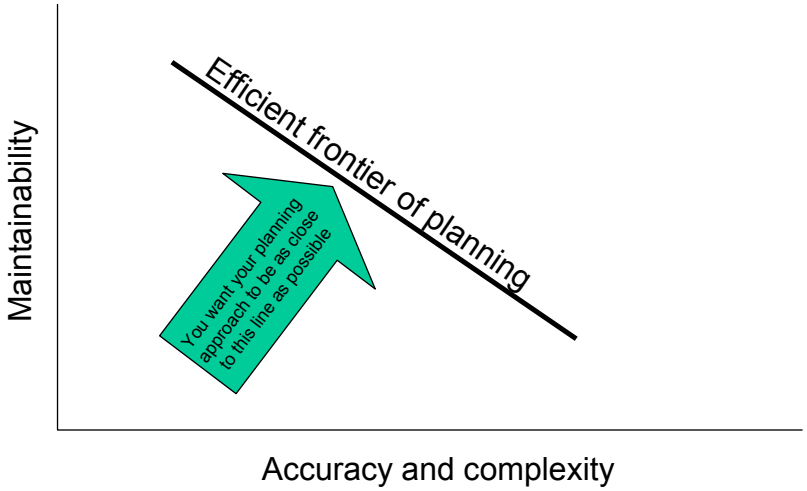
But regardless of the form of the model, it is a simplification of reality. It does not include where all the molecules of our product are at present, or even (to be a little more practical) precisely where all the boxes are as of any given moment of time. If it is a tactical planning modeling designed to look months into the future, it probably deals with groups of products rather than all the unique SKUs we could identify. If it is the model built into the operating database of a warehouse management system, it is much more detailed (typically SKUs and specific storage locations) but still far from a complete representation of reality.

As we mention in the book, the "model is always wrong". This is true both because planning models project somewhat into an always uncertain future, and because they don't capture all the detail. We can make a model more accurate by making it more detailed, but at the cost of making it more difficult to comprehend and to maintain. And as models get more complex and harder to understand, we (and our management) begin to lose trust that they are correct.

There is a kind of fundamental tradeoff here, as shown by the sloping line in the figure below, with increased accuracy and complexity and of a model leading to reduced maintainability and greater planning costs. This tradeoff is also one of the drivers behind the universal appeal of hierarchical planning, i.e. the use of more aggregate models to plan major decisions further into the future, and more detailed models to make more specific decisions in the near term: to try to keep the total amount of detail and information flowing through the models to the minimum necessary to do effective planning.

But the figure also shows another aspect of this challenge. Not only do we have the tradeoff between accuracy and maintainability, but we have the challenge of trying to be close to the "efficient frontier" of that tradeoff, a concept borrowed from micro-economics. What do we mean by being close to the efficient frontier? Like most uses of the word efficiency, it means "getting the most out of the process for the least put into the process." Specifically, it means getting the most planning value out of the complexity and data maintenance effort put into it. It means *elaborating our planning model where that elaboration produces real planning value, and keeping it simple where we don't need the detail.*

Topic 3, Figure 1: Accuracy tradeoff of plans



It's easy to have a model that is not near this efficient frontier, that has modeling detail all over the place regardless of its value. For example If you need to make decisions about warehouse capacity next year, you can take the operating database of a warehouse management systems, create a simulation scenario from it, increase the numbers of orders across the board, schedule operations for some simulated weeks next year, and see where you have capacity constraints. It works, and you may learn a lot, but it is a ton of work to do that simulation of next year. If all we are trying to do is project labor force and check the number of dock doors and square footage, there are much simpler models we can use to produce reasonably accurate projections.

Usually the right approach to model building is to consider the decisions that have to be made, build the simplest model that we dare, and then experiment with planning using it. Then, as we have problems with the model glossing over certain key issues, we add complexity where it is needed. Thus the structure of the model becomes a "tuning" parameter for our planning (see Chapter 8 of the book for a discussion of tuning).

Suppose we have a capacity-based model that is supporting our master scheduling. We believe that the right level of detail is modeling manufacturing cells as complete units that can, say, "make this many of this family of components per hour". But we discover after a few months of use that for one of our cells with a wide range of parts sizes, we keep getting into capacity trouble with our master schedule because the cell really has two presses in it, one for large parts and one for small, and if we want to spend a

week building only product that uses only the large parts made by the one press, we really don't have as much capacity as we thought. It's time to break the cell down in our model, model the two types of capacity separately, and start working with more realistic, slightly more complex models.

One of the problems that finite capacity detail scheduling tools have traditionally had is that it is sometimes quite difficult to find any point near that accuracy versus complexity tradeoff that actually works practically. If we build a model with a lot of shop complexity in it, we have created a terrible maintenance bear that we keep being tempted to drop because we are spending hours per week keeping it accurate. If we keep it simple, we get proposed schedules out of the system that are so far from realistic that we spend as much time editing them into make-able schedules as if we had manually started from scratch – and maybe not of any higher quality. This is, by no means, a universal problem, but it happens in a significant fraction of cases. And of course, we remain tormented by not knowing whether, if we were better modelers, we could have created a more efficient model and now be using it happily.

Our guideline remains, start as simple as we think captures the essence, and add complexity where required.