Production Scheduling as a Gateway to Smart Manufacturing for SMEs.

Peter Denno, NIST
2019-06-04
Message of the Presentation

• Production scheduling is hard because scheduling requirements differ greatly and change frequently.

• SMEs could improve their schedules were they provided a gentle introduction to the use of sophisticated analytical tools.

• Starting with their familiar scheduling spreadsheets, there is such a gentle path, and it can introduce them to Web APIs too.
Scheduling. Hard.
Requirement, constraints, and goals from many sources
Scheduling Problem Formulation

3 high-level goals

1. Describe system capacity.
2. Describe the work to be done.
3. Describe what is important about how the work is done.
Scheduling Problem Formulation

3 high-level goals

1. Describe system capacity.
2. Describe the work to be done.
3. Describe what is important about how the work is done.
Scheduling Problem Formulation

3 high-level goals

1. Describe system capacity.
2. Describe the work to be done.
3. Describe what is important about how the work is done.
Scheduling Problem Formulation

3 high-level goals

1. Describe system capacity.
2. Describe the work to be done.
3. Describe what is important about how the work is done.

Any of the inputs plus customer priorities
Today SMEs typically schedule production with very little IT decision support.

But scheduling could be their gateway to smart manufacturing.

How?

It starts with their production spreadsheets.

And we are exploring it with industrial pilots.
Assumptions about Industrial Pilot Partners

• We assume that if you *even do have* a scheduling system, its name ends in `.xlsx`
  • But that’s a good start!

• You are an SME, or able to work agile and hands-on like one.

• At least one person, 4 hours/week, willing to learn, brave and true.

• Willing to share your experience / questions with other SME manufacturers
  • We’ll set up a blog and google group etc.
  • Absolutely no obligation to share anything you might view as a specific competitive advantage
Willing to learn what?
Two Skill-level Weighted-penalty Scheduling

Overview

This notebook calculates a schedule in which the penalty of delivering orders late is minimized. The penalty weights indicate how significant being late is by line (which assume means also "by customer"). For example, a low-priority customer weight could be 1, and being 3 weeks late on an order for that customer would produce a penalty of 3. A high-priority customer weight could be 5, and being 3 weeks late on an order for that customer would produce a penalty of 15.

Definitions (thing you know but I had to learn)

finished garment item number: A garment product type identifier that provides size and color. Example F.G. Item Numbers are "6507-NY705-2XL". Note that the first four digits of a finished garment item number are its route or routing number. The route of the example is 6507.

line: A garment product type identifier that ignores size and color. Example lines are [NB T], and [PTGN L9 PNT]. The various finishd garment types (identified by finished garment item numbers) all are constructed the same and take the same amount of time to make.

routing number: (AKA ‘route’) A number that identifies a garment product type possibly more specific than a line. For example, the routing number 6915 and 6891 both designate MARS W00L SS SHT. Note: The route is almost always a
Step 3: Find Size of Workforce and Summarize Skills Data

The thing created below, \texttt{skills}, is a (Python) Pandas DataFrame.

\begin{verbatim}
[9]:
  skills = pd.read_excel("Data/skills-3column.xlsx", sheet_name="Sheet1", usecols=[0,1,2])
  skills.sample(3)
\end{verbatim}

<table>
<thead>
<tr>
<th>EmployeeID</th>
<th>Location</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>174</td>
<td>15563</td>
<td>Greenhouse Quality Inspector, Advanced Level</td>
</tr>
<tr>
<td>598</td>
<td>31165</td>
<td>Greenhouse Apprentice Sewer Level 1</td>
</tr>
<tr>
<td>126</td>
<td>13254</td>
<td>Greenhouse Quality Inspector, Basic Level</td>
</tr>
</tbody>
</table>

We will only be scheduling the bottleneck process (sewing), so let's remove everyone who isn't a sewer. First, let's look at what the different values in the \texttt{Title} column are. This is done with the Pandas \texttt{unique} function.
% This model lets jobs be late but penalizes them for it.
% FEATURES:
  1) Two skill levels.
  2) WorkdaysPerWeek and WorkhoursPerDay (for whole weeks).
  3) Keeps the number of workers the line constant.
  4) Balances line by ratio of high-skilled to total.
  5) Penalizes late jobs with customer-priority weighted penalties.
% PROs: Doesn't require what-ifs to find a satisfactory solution.
% CONs: It seems not to find a solution when minimizing on penalty. (Set penalty manually)

int: workforceSize;
int: maxTeamsOnJob;
int: numLines;
set of int: Lines = 1..numLines;
array[Lines] of int: LinePenalty;

int: numJobs;
set of int: Jobs = 1..numJobs;
array[Jobs] of Lines: LineOfJob;

int: schedulingWeeks;
set of int: Weeks = 1..schedulingWeeks;
array[Weeks] of int: WorkdaysPerWeek;
array[Weeks] of int: WorkhoursPerDay;

array[Lines] of int: TeamLowsByLine;
array[Lines] of int: TeamHighsByLine;

% RequiredWorkEffort is the number of sewer/days needed to complete the job.
% It is lhpq times quantity.
How do Web APIs fit in?

• At least medium-sized manufacturers typically have bumped up against the limitations of using spreadsheets.
• They may have an ERP system.
• Step 2 in their journey is to stop loading spreadsheets and start making API calls.
  • We have in mind a tool for this....
API Prototyping from Spreadsheets

- Provide a tool that reads from spreadsheets and serves the same data using a Web API.
- Training wheels for Web API
- Try it with your ERP, customer, or supplier.
POST /math/plus

Response Class (Status 200)

Model  | Example Value

Response Content Type  | application/json

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Parameter Type</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>body</td>
<td>Model</td>
</tr>
</tbody>
</table>

Parameter content type:

- application/edn

```
inline_model_1{
  x (integer),
  y (integer)
}
```
More time?
Scheduling Problem Formulation
3 high-level goals

1. Describe system capacity.
2. Describe the work to be done.
3. Describe what is important about how the work is done.

• A few examples of subgoals:
  • Make description of (2) compatible with (1).
  • Divide (3) into constraints and objective.
  • Find relationship types (functional, injective etc.)
  • Match relationships to global constraints
Plan Recognition (typical)

- Hierarchical plan library capable of expressing very many possible plans (1000s).
  - Plans express a reasonable set of user goals
- Observations about what the user is doing suggest some goals as more likely.
- Probabilistic (Hidden Markov Model)
- Build candidate plans lazily, bottom up.
- Knowing the user’s plan helps agent act.
Plan Recognition (using an analytical tool)

- Plan lib decomposes the 3 high-level goals
  - Plans express a reasonable set of user goals
- Observations are about the meaning of data and how the user seems to be using it.
- Knowing the user’s plan helps agent validate what the user is doing (e.g. in the MiniZinc)
System Architecture

MiniZinc Solver

Notebook & User

"Ontology Server"
Consensus terms
Specific terms
API Definitions

New terms
Term definitions
NBJSON
Advice

Advisor App
(Plan Recognizer)

Explanation Editor
Explanation Objects

Ipython magic

Mzn MM = Minizinc Meta-Model
Conclusion

• If SMEs are to benefit from smart manufacturing, they will need some help.
• We have an initiative to provide that help,
• And it can involve Web APIs and someday maybe messaging standards.
Thanks!
Why engage SMEs?

• Hands-on
• Multi-skilled
• Incentivized
• Might not have much automation
• Might not have deep computer skills
  • But use Excel a great deal
• Perfect!
MiniZinc

• A language for describing combinatorial optimization problems
• Binds to many solvers

```plaintext
constraint forall (w in Weeks) (sum (j in Jobs) (WorkersOnJob[j,w]) <= workforce_size);

% Work on a job starts before it ends!
constraint forall (j in Jobs) (startWeek[j] <= endWeek[j]);

% Work on a job ends by its due date.
constraint forall (j in Jobs) (endWeek[j] <= WeeksTillDue[j]);
```
Summary of Part2
(engineering such a system)

- Jupyter Notebooks
  - Embedded MiniZinc cells to solve the combinatorial problem
- Model of explanation supporting notebook (goals, causality)
  - Obtained through NLP and manual edits
  - Linked to code, ontology terms
  - Checked with HTN planning tool
  - Updated immutably (Immutable DB with Datalog interface)
- MS Clippy-like agent monitors notebook and explanation
- Ontology links allow participants to share ideas, not rely on the magic of one individual.
Peckham Scheduling Notebook

• For production scheduling and ”what if” analysis
• Dissect and clean data from Excel spreadsheets with Pandas
  • Orders, due dates
  • Capacity: labor quantity and skills
  • Products: labor hours per garment, skills required
• 60 lines of MiniZinc embedded in notebook
• Objective /Constraints
  • Combination line balancing / scheduling
  • Minimize change-over
  • Meet due dates
• Leave material usage constraint as “an exercise for the user”
• Explanation metamodel not yet integrated
Backup Slides
Design Challenge 1: Insightful use of solvers
A spreadsheet is a set of sentences.

- **array [Jobs] of var 1..weeksScheduled: startWeek;**

- First, let’s consider what we are trying to represent here. What this is saying would be said in Excel using a table with two columns: One column would be “Job ID,” the other would be “Start Week”. In an Excel table like that, the Job IDs are the topics of sentences involving the other column, the start weeks.

- In Excel, column headings tell you the form of sentences, typically one of the columns is the topic.

- Minizinc goes about doing the same thing a little differently. In one statement it says what the topic of some sentences will be, then in other statements it lets you define the form of sentences.

- Here is how you say that a set of jobs, called Jobs can be the topic of sentences: set of int: Jobs = 1..numJobs;

- Let’s drill down into that.