Rescheduling Bulk Gas Production and Distribution

Wasu Glankwamdee
Jackie Griffin
Jeff Linderoth
Jierui Shen

Andy Bringhurst
Jim Hutton

Brian Baumrucker
Larry Biegler

PITA EWO Meeting Carnegie Mellon March 15, 2006
Liquid Bulk Gas Production-Distribution

- Sites $S$
- Products $\mathcal{P} = \{\text{LOX, LNI}\}$
- Customers $\mathcal{C}$

Planning Problem
How should one set production levels at the sites $s \in S$ and sourcing decisions (amount delivered from $s \in S$ to $c \in \mathcal{C}$) in order to meet customer demand at minimum cost?
Bulk Gas Wrinkles

**Production**
- Most sites operate in two modes:
  - Regular Mode
  - Extended Mode (Costs more than regular)
- Physics of Production
  - Maximum total production: \((\text{LOX} + \text{LNI})\)
  - Individual production limit. (Fraction of total)

**Competitor Arrangements**
- Enter contractual “take-or-pay” arrangements with competitors.
- Allowed to remove (equal) fixed amount of product from each other’s sites
  - \(Q \subseteq S\) : Set of ‘Pick up’ locations
  - \(R \subseteq S\) : Set of ‘Take out’ locations
A Simple Planning Model

Objective

\[ \min \sum_{p \in \mathcal{P}} \sum_{s \in \mathcal{S}} (\alpha_{ps} x_{ps} + \beta_{ps} e_{ps}) + \sum_{p \in \mathcal{P}} \sum_{s \in \mathcal{S}} \sum_{c \in \mathcal{C}} (d_{sc} y_{psc}) \]

Variables

- \( x_{ps} \): Regular production amount of \( p \in \mathcal{P} \) at \( s \in \mathcal{S} \)
- \( e_{ps} \): Extended production amount of \( p \in \mathcal{P} \) at \( s \in \mathcal{S} \)

Parameters

- \( \alpha_{ps} \): Regular mode per unit production cost of \( p \in \mathcal{P} \) at \( s \in \mathcal{S} \)
- \( \beta_{ps} \): Extended mode per unit production cost of \( p \in \mathcal{P} \) at \( s \in \mathcal{S} \)
- \( d_{sc} \): per-unit delivery cost from \( s \in \mathcal{S} \) to \( c \in \mathcal{C} \)
Constraints

**Maximum Production Level**

\[ \sum_{p \in \mathcal{P}} x_{ps} \leq M_s, \quad \sum_{p \in \mathcal{P}} e_{ps} \leq N_s \quad \forall s \in S \]

\[ x_{ps} \leq \Lambda_p M_s, \quad e_{ps} \leq \Lambda_p N_s \quad \forall p \in \mathcal{P}, \forall s \in S \]

- **Parameters**
  - \( M_s, N_s \): Regular mode maximum total production at \( s \in S \)
  - \( N_s \): Extended mode maximum total production at \( s \in S \)
  - \( \Lambda_P \): Maximum “air-fraction” of \( p \in \mathcal{P} \)

- This is a fairly crude (approximate) model of production
Constraints

**Contract Amount Limit**

\[ \sum_{q \in Q} x_{pq} \leq \Phi_p \quad \forall p \in P \]

**Customer Demand**

\[ \sum_{s \in S} y_{psc} \geq B_{pc}, \quad \forall p \in P, \quad \forall c \in C \]

- **Variables**
  - \( y_{psc} \): Amount of \( p \in P \) shipped from \( s \in S \) to \( c \in C \)

- **Parameters**
  - \( \Phi_p \): Contract amount for \( p \in P \)
  - \( B_{pc} \): Customer \( c \in C \) demand for \( p \in P \)
Constraints

Inventory Balance

\[ x_{ps} + e_{ps} - \sum_{c \in C} y_{psc} - z_{ps} = \Delta I_{ps}, \quad \forall p \in \mathcal{P}, \forall s \in S \]

Resource: Driver Hours and Truck Hours

\[ \sum_{p \in \mathcal{P}} \sum_{c \in \mathcal{C}} d_{sc} y_{psc} \leq D_s, \quad \forall s \in S \]
\[ \sum_{c \in \mathcal{C}} d_{sc} y_{psc} \leq K_{ps}, \quad \forall p \in \mathcal{P}, \forall s \in S \]

- **Variables**
  - \( \Delta I_{ps} \): Change in inventory of \( p \in \mathcal{P} \) at \( s \in S \)

- **Parameters**
  - \( z_{ps} \): Amount of \( p \in \mathcal{P} \) competitor removes from \( s \in S \)
  - \( D_s \): Available driver hours at \( s \in S \)
  - \( K_{ps} \): Available truck hours of \( p \in \mathcal{P} \) at \( s \in S \)
Using the Production-Distribution Model

- Model is used to set *monthly production levels* and *customer sourcing decisions*
- Sometimes, during the course of the month, things get “out of skew”
  - A customers is about to be run out
  - Plants don’t have enough product to meet short-term customer demand

**What Happens in Practice**

- Daily planners (attempt) to do a manual adjustment to the monthly schedule in order to meet customer demand
- Sometimes, the planning model will be re-run given the current (changed) input conditions.
Why?

One Possible Explanation

**Known** variation in plant supply and customer demand during the course of the month

- **Customer Usage**
- **Aggregate/Prorated Delivery Volume**
- **True Customer Inventory**

What’s the Cure!?

The rescheduling burden can be alleviated by solving the model at **a finer time aggregation**
Why?

Another Possible Explanation

**Unknown** (random) variation in components of the model

- A plant just went down for an unscheduled maintenance.
- A customer used three times as much as forecast

What’s the Cure?

- In that case, a planning model that **directly** deals with the inherent uncertainty might be warranted
  - Stochastic Programming
  - Robust Optimization
What to Do?

- In either case, building and implementing a new model within APCI is not to be taken lightly.

Project Scope

- The company is interested in understanding why this rescheduling must often take place

Step #1

- Build and experiment with a multi-period version of the planning model.
- Will allow us to experiment with instances solved at variety of time grains
- Would need a multi-period model in order to create a stochastic program anyway
A Multi-Period Planning Model

- \( T = \{1, 2, \ldots, T\} \): Set of time periods
- Essentially add a “time index” to all the variables and parameters
- Make inventory a variable that can be carried from one period to the next, e.g.

\[
x_{p s t} + e_{p s t} - \sum_{c \in C} y_{p s c t} - z_{p s t} + I_{p r, t-1} - I_{p r t} = 0 \quad \forall p \in \mathcal{P}, \forall r \in \mathcal{R}, \forall t \in T
\]

- Also add inventory cost to objective

\[
\sum_{p \in \mathcal{P}} \sum_{s \in \mathcal{S}} \sum_{t \in T} \gamma_{p s} I_{p s t}
\]
Creating the Model

- Model built and created in **Mosel** modeling language
- Mosel is **convenient**
  - Air Products uses Mosel
  - If we wish to build a stochastic programming model later on, Mosel has new modules for building stochastic programs
- Model reads instance data from (properly formatted) text files
The Facts

1. Models need data
2. Data is hard to come by
3. ⇒ We will create our own data.

- Instance Generation-Simulation code being created in C++
- Data is random, but reasonable

**Sites**
- Daily Production Rate
- with random outages

**Customers:**
- Normally Distributed
- On-Off
- Call-in
Class Structure

- **Site**
  - Location
  - DailyMaxProduction
  - SiteProductInfo (NumTrucks, Initial Inventories, etc.)
- **Customer**
  - Location
  - Product
  - (Abstract) DemandDistribution Class
Class Structure

- **InstanceFamily**
  - NumDays
  - Sites
  - Customers

- **Instance**
  - Something that can be solved!
  - `create(InstanceFamily &if, vector<int> &daysPerPeriod)`

- All classes have a `makeRandom()` method that will instantiate itself with random, reasonable, data.

- When data is available, we can extended classes to instantiate themselves by reading from a file.
int main(int argc, char *argv[]) {
    InstanceFamily testInstance;

    // 3 sites, 12 customers, 10 days
    testInstance.makeRandom(3, 12, 10);

    Instance instance;
    vector<int> daysPerPeriod(4);
    daysPerPeriod[0] = 2;
    daysPerPeriod[1] = 2;
    daysPerPeriod[2] = 3;
    daysPerPeriod[3] = 3;

    instance.create(testInstance, daysPerPeriod);
    instance.writeMosel('EWO_AP.dat');
}
Experiments

Experiment #1 — Can We?

- Build large multi-period models
- See if state-of-the-art commercial solvers (XPRESS/CPLEX) as well as open-source solvers (Clp, GLPK) can solve instances in reasonable computing time

Experiment #2 — What do we gain?

1. Solve same InstanceFamily with large and small time buckets
2. Simulate customer inventories
   - Deliveries made in equal (daily) increments
   - Customer demand varies daily
Experiments

**Metrics**

1. Total number of customer outages
2. Average customer outage amount
3. Others?

**Other Experiments**

1. Measure cost and benefit of solving for a “robust” solution in which customer demand is slightly exceeded
2. How to handle “competitor relationships”
   - As a Parameter?
   - As a Variable?
   - Random Variable!
Results

Hopefully Some Preliminary Results by 3/15!