A Novel Global Optimization Approach to the Multiperiod Blending Problem

Scott Kolodziej
Advisor: Dr. Ignacio Grossmann

Department of Chemical Engineering
Carnegie Mellon University
Problem Definition

Given Information

- **Time Horizon (Periods)**
- **Network Topology**
- **Supply Tanks**
  - Amount entering
  - Concentrations
- **Demand Tanks**
  - Amount withdrawn
  - Concentration limits
- **Initial Conditions**
  - Inventories
  - Concentrations
- **Economic Costs**
  - Network flow costs
  - Raw material costs
  - Profit for meeting demand

Determine

- Flows between which tanks in which time periods
- Inventories and concentrations for all tanks in each time period
- **Maximum total profit** of blending operation
Mathematical Model

Objective: Maximize Profit

- Mass Balances
  - Overall Flows
  - Individual Components (Blending)
- Flow/Inventory Bounds
- Operational Constraints
- Demand Specifications

Variables
- Flows, Concentrations, and Inventories (Continuous)
- Existence/Nonexistence of Streams (Binary)

Complicating nonconvex bilinearities $F \cdot C$ and $I \cdot C$ appear in the individual component mass balances. Requires global optimization techniques. Resulting model is an MINLP.
Radix-Based Discretization

Given that the bilinear product $u = F \cdot C$ can be replaced by this set of constraints using exact linearization:

$$C = \sum_{k=p}^{P} \sum_{j=0}^{9} 10^k \cdot j \cdot z_{j,k}$$

$$u = \sum_{k=p}^{P} \sum_{j=0}^{9} 10^k \cdot j \cdot \hat{F}_{j,k}$$

$$\hat{F}_{j,k} \leq F^U \cdot z_{j,k} \quad \forall \ k \in \{p, \ldots, P\}, j \in \{0, \ldots, 9\}$$

$$\sum_{j=0}^{9} \hat{F}_{j,k} = F \quad \forall \ k \in \{p, \ldots, P\}$$

$$\sum_{j=0}^{9} z_{j,k} = 1 \quad \forall \ k \in \{p, \ldots, P\}$$

$$z_{j,k} \in \{0, 1\} \quad \forall \ k \in \{p, \ldots, P\}, j \in \{0, \ldots, 9\}$$

F is disaggregated but still continuous

Discretization replaces bilinearity with mixed-integer linear constraints

The MINLP model can be reformulated as an MILP approximation

Teles, Castro, & Matos (2011)
Radix-Based Discretization

Radix (or base) $k$ selects over powers of 10

0-1 Binary Variable (one for each digit)

$j$ selects over set of digits

Ensures only one digit for each decimal place

$C = \sum_{k=p}^{P} \sum_{j=0}^{9} 10^k \cdot j \cdot z_{j,k}$

$\sum_{j=0}^{9} z_{j,k} = 1 \quad \forall k \in \{p, \ldots, P\}$

C Axis

<table>
<thead>
<tr>
<th>Increment</th>
<th>1</th>
<th>0.1</th>
<th>0.01</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>0-9</td>
<td>0-9.9</td>
<td>0-9.99</td>
</tr>
<tr>
<td>Significant Digits</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Binary Variables</td>
<td>10</td>
<td>20</td>
<td>30</td>
</tr>
</tbody>
</table>

Linear increase in binary variables for order of magnitude increase in precision

Teles, Castro, & Matos (2011)
Lower Bounding

• RBD yields an **upper bound**
• What about a **lower bound**?

We introduce $\Delta C$ as a slack variable

$\Delta C$ ranges between 0 and $10^p$ (the size of the “gaps”)

Relaxed RBD constraints can be added to “fill the gap” and relax the original problem
Lower Bounding

- Feasible region is identical to piecewise McCormick Envelopes
- Relaxation is less tight

Feasible region is now relaxed
Upper and Lower Bounding

Upper Bounding

Lower Bounding (Relaxation)

Underestimates the feasible region

Overestimates the feasible region
Global Optimization Algorithm

- As precision increases, $UB \rightarrow LB$
- We can start with low precision and slowly increase precision until $\text{gap} < \varepsilon$
- Currently work in progress
Global Optimization Algorithm

- Preliminary Computational Results

<table>
<thead>
<tr>
<th>Tanks</th>
<th>6</th>
<th>8</th>
<th>8</th>
<th>8</th>
<th>8</th>
<th>8</th>
<th>8</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Periods</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Global Optimization Wall Time (s)</td>
<td>2.1</td>
<td>4753</td>
<td>2588</td>
<td>426</td>
<td>338</td>
<td>&gt;7200</td>
<td>1525</td>
<td></td>
</tr>
<tr>
<td>MINLP Wall Time (s)</td>
<td>21.2</td>
<td>&gt;7200</td>
<td>&gt;7200</td>
<td>&gt;7200</td>
<td>&gt;7200</td>
<td>&gt;7200</td>
<td>&gt;7200</td>
<td></td>
</tr>
<tr>
<td>Relative Gap</td>
<td>0</td>
<td>5.19x10^{-4}</td>
<td>1.60x10^{-3}</td>
<td>0</td>
<td>6.86x10^{-4}</td>
<td>2.34x10^{-3}</td>
<td>3.39x10^{-3}</td>
<td></td>
</tr>
<tr>
<td>Iterations</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

Generally much faster than MINLP
Conclusions

• MINLP approach takes **too long**
• Effective **radix-based MILP approach** reviewed and expanded
  – **Order of magnitude reductions** in CPU time observed
• **Upper and lower bounds** were introduced using **radix-based discretization**

Future Work

• Utilize **upper/lower bounding** formulations to **globally optimize**
• Large scale instances may require **decomposition techniques**
  – Utilize **Lagrangian decomposition** to decompose larger problems into easier to solve **subproblems**