NEW MULTI-COMMODITY FLOW FORMULATIONS FOR THE GENERALIZED POOLING PROBLEM

Fabian Rigterink\textsuperscript{a}
Natashia Boland\textsuperscript{b}, Thomas Kalinowski\textsuperscript{a} and Martin Savelsbergh\textsuperscript{b}

July 16, 2015

\textsuperscript{a} The University of Newcastle, Australia · School of Mathematical and Physical Sciences
\textsuperscript{b} Georgia Institute of Technology · H. Milton Stewart School of Industrial and Systems Engineering
1. Introduction
2. Formulations
3. Computational results
4. Conclusion
5. An application
INTRODUCTION
The pooling problem:

Given a set of raw material suppliers (inputs) and qualities of the supplies, find a cost-minimising way of blending these raw materials in intermediate pools and outputs so as to satisfy requirements on the output qualities.
• Numerous equivalent formulations for the pooling problem have been proposed, e.g. the P-, Q-, PQ- and HYB-formulations, and recently a multi-commodity flow formulation based on input commodities.

• An important characteristic of a formulation is the degree of variable disaggregation, which comes at a cost: the higher the degree of variable disaggregation, the larger the problem.

• Formulations with a low degree of variable disaggregation often perform poorly, but so do formulations with a high degree of variable disaggregation.

→ An interesting trade-off to study!
The main contributions of our paper:

1. We introduce new multi-commodity flow formulations for the pooling problem,
The main contributions of our paper:

1. We introduce new multi-commodity flow formulations for the pooling problem,

2. we prove their equivalence and present a description of the partial order of formulations with respect to the strength of the LP relaxations, and
The main contributions of our paper:

1. We introduce new multi-commodity flow formulations for the pooling problem,
2. we prove their equivalence and present a description of the partial order of formulations with respect to the strength of the LP relaxations, and
3. we evaluate the performance of the new formulations in a computational study to find a good degree of variable disaggregation.
Let $G$ be a directed graph where

$$G = (V, A) \quad V: \text{vertices}, A: \text{arcs}$$

$I, L, J \subseteq V \quad I: \text{inputs}, L: \text{pools}, J: \text{outputs}$

**Assumption:**

$$A \subseteq (I \times L) \cup (L \times L) \cup (L \times J) \cup (I \times J)$$

**Terminology:**

$L = \emptyset \quad \Rightarrow \quad \text{blending problem (LP)}$

$A \cap (L \times L) = \emptyset \quad \Rightarrow \quad \text{standard pooling problem}$

$A \cap (L \times L) \neq \emptyset \quad \Rightarrow \quad \text{generalized pooling problem (both NLPs)}$
FORMULATIONS
**KNOWN FORMULATIONS**

All formulations

\[ y_a \quad \text{Flow on arc } a \in A \]

P-formulation

\[ p_{vk} \quad \text{Quality value of } v \in I \cup L \]

for quality \( k \in K \)

\[ (p_{vk} \equiv \lambda_{vk}, \ v \in I, \ k \in K) \]
Inputs: MCF-I-(P)Q-formulation

$q^l_{iv}$  Fraction of total incoming/outgoing flow of $v \in I \cup L$ that comes from input $i \in I$

$x^l_{ia}$  Flow in $y_a$, $a \in A$, that comes from input $i \in I$
\[\text{MIN: MCF-I-Q-FORMULATION}\]

\[
\begin{align*}
\text{min} & \quad q^i, x^i, y \\
& \sum_{a \in A} c_a y_a \\
\text{s.t.} & \\
& \sum_{a \in \delta^{\text{in}}(v)} y_a = \sum_{a \in \delta^{\text{out}}(v)} y_a, & v \in L, \quad (1) \\
& \sum_{a \in \delta^{\text{out}}(v)} y_a \leq C_v, & v \in I \cup L, \quad (2) \\
& \sum_{a \in \delta^{\text{in}}(v)} y_a \leq C_v, & v \in J, \quad (3) \\
& 0 \leq y_a \leq u_a, & a \in A, \quad (4) \\
& q^i v \geq 0, & v \in I \cup L, \quad (5) \\
& \sum_{i \in I} q^i v = 1, & v \in I \cup L, \quad (6) \\
& \sum_{a \in \delta^{\text{in}}(v)} x_{ia}^i = \sum_{a \in \delta^{\text{out}}(v)} x_{ia}^i, & v \in L, \quad i \in I, \quad (7) \\
& \sum_{i \in I} \sum_{a \in \delta^{\text{in}}(j)} \lambda_{ik} x_{ia}^i \geq \mu_{jk}^{\text{min}} \sum_{a \in \delta^{\text{in}}(j)} y_a, & j \in J, \quad k \in K, \quad (8) \\
& \sum_{i \in I} \sum_{a \in \delta^{\text{in}}(j)} \lambda_{ik} x_{ia}^i \leq \mu_{jk}^{\text{max}} \sum_{a \in \delta^{\text{in}}(j)} y_a, & j \in J, \quad k \in K, \quad (9) \\
& x_{ia}^i = q_{ia} y_a, & a \in A, \quad i \in I. \quad (10)
\end{align*}
\]
Inspired by the ideas of the Reformulation-Linearization Technique [6], the MCF-I-PQ-formulation adds valid (but redundant) constraints to the MCF-I-Q-formulation.

\[
\begin{align*}
\min_{q^I, x^I, y} \quad & \sum_{a \in A} c_a y_a & \quad \text{s.t.} & \quad (1) - (10), \\
\end{align*}
\]

\[
y_a = \sum_{i \in I} x_{ia}^l, \quad a \in A, \quad (11)
\]

\[
\sum_{a \in \delta_{out}(v)} x_{ia}^l \leq C_v q_{iv}^l, \quad v \in L, \quad i \in I. \quad (12)
\]
Outputs: MCF-J-(P)Q-formulation

\( q_{jv}^I \) Fraction of total incoming/outgoing flow of \( v \in L \cup J \) that goes to output \( j \in J \)

\( x_{ja}^I \) Flow in \( y_a, a \in A \), that goes to output \( j \in J \)
Inputs + outputs: MCF-(I+J)-(P)Q-formulation
\[ q_{iv}, \ x_{ia}, \ q_{jv}, \ x_{ja} \quad \text{See above} \]

Inputs × outputs: MCF-(I×J)-(P)Q-formulation
\[ q_{iv}, \ x_{ia}, \ q_{jv}, \ x_{ja} \quad \text{See above} \]
\[ q_{ija}^{ij} \quad \text{Fraction of flow } y_a, \ a \in A, \text{ that comes from input } i \in I \text{ and goes to output } j \in J \]
\[ x_{ija}^{ij} \quad \text{Flow in } y_a, \ a \in A, \text{ that comes from input } i \in I \text{ and goes to output } j \in J \]
Table 1: Known and new formulations for standard pooling problems (SPP) and generalized pooling problems (GPP)

<table>
<thead>
<tr>
<th>Formulation</th>
<th>SPP</th>
<th>GPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCF-I-(P)Q</td>
<td>known [3, 7]</td>
<td>known [1]</td>
</tr>
<tr>
<td>MCF-J-(P)Q</td>
<td>known [2]</td>
<td>new</td>
</tr>
<tr>
<td>MCF-(I+J)-(P)Q</td>
<td>known [2]</td>
<td>new</td>
</tr>
<tr>
<td>MCF-(I×J)-(P)Q</td>
<td>new</td>
<td>new</td>
</tr>
</tbody>
</table>
All edges are directed upwards. A solid edge \((f_1, f_2)\) indicates \(f_2 \succeq f_1\). Dotted edges indicate non-edges, i.e., two formulations are incomparable. A dashed edge indicates a conjectured solid edge.

Figure 1: Hasse diagram of the partial order of formulations.
**STRENGTH OF FORMULATIONS**

Edges:
See [1], Proposition 2.

Figure 1: Hasse diagram of the partial order of formulations.
STRENGTH OF FORMULATIONS

Edges (cont.):
Any PQ-formulation is at least as strong as its corresponding Q-formulation.

Figure 1: Hasse diagram of the partial order of formulations.
STRENGTH OF FORMULATIONS

Edges (cont.):
Any (I×J)-formulation is at least as strong as its corresponding (I+J)-formulation.

Figure 1: Hasse diagram of the partial order of formulations.
Edges (cont.):
Any (I+J)-formulation is at least as strong as both its corresponding I- and J-formulations.

Figure 1: Hasse diagram of the partial order of formulations.
Non-edges:
Proof by altering instance gppL1.

Figure 1: Hasse diagram of the partial order of formulations.
Conjectured edge:
Our computational results suggest $\text{MCF-(I+J)} \text{-PQ} \succeq \text{MCF-(I\times J)} \text{-Q}$.

Figure 1: Hasse diagram of the partial order of formulations.
COMPUTATIONAL RESULTS
We investigated 74 pooling problem instances

- 34 of which are SPP instances and
- 40 of which are GPP instances.

From easy instances with 6 vertices and 6 arcs (Haverly1)...

...to hard instances with 130 vertices and 1451 arcs (sppC3)...

![Graph with 6 vertices and 6 arcs]

![Complex graph with many arcs]
• We do a McCormick relaxation [5] of the bilinear term $z = xy$ on $x \in [x^L, x^U]$ and $y \in [y^L, y^U]$:  

$$
\begin{align*}
  z &\geq xy^L + x^L y - x^L y^L, \\
  z &\leq xy^L + x^U y - x^U y^L,
\end{align*}
\begin{align*}
  z &\geq xy^U + x^U y - x^U y^U, \\
  z &\leq xy^U + x^L y - x^L y^U.
\end{align*}
$$

• We solve all relaxed linear programs with CPLEX 12.6.0.0 and all nonlinear programs with SCIP 3.0.0.

• The nonlinear programs run with a time limit (time_limit) of 600 seconds.
Table 2: Absolute size of formulations.

<table>
<thead>
<tr>
<th></th>
<th>nbils</th>
<th>LP_nvars</th>
<th>LP_ncons</th>
<th>LP_nnonzeros</th>
<th>NLP_nvars</th>
<th>NLP_ncons</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>P</td>
<td>Q</td>
<td>PQ</td>
<td>Q</td>
<td>P</td>
<td>PQ</td>
</tr>
<tr>
<td>MCF-I-</td>
<td>5617</td>
<td>1158</td>
<td>1127</td>
<td>2285</td>
<td>7918</td>
<td></td>
</tr>
<tr>
<td>MCF-J-</td>
<td>11042</td>
<td>7006</td>
<td>6900</td>
<td>9490</td>
<td>23729</td>
<td></td>
</tr>
<tr>
<td>MCF-(I+J)-</td>
<td>24084</td>
<td>6784</td>
<td>6564</td>
<td>12956</td>
<td>48357</td>
<td>51960</td>
</tr>
<tr>
<td>MCF-(I×J)-</td>
<td>26401</td>
<td>40155</td>
<td>39865</td>
<td>79440</td>
<td>127230</td>
<td>100085</td>
</tr>
<tr>
<td>P</td>
<td>5425</td>
<td>5849</td>
<td>5773</td>
<td>7205</td>
<td>15811</td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td>1616</td>
<td>2154</td>
<td>2453</td>
<td>2055</td>
<td>3817</td>
<td>16685</td>
</tr>
<tr>
<td>PQ</td>
<td>20288</td>
<td>20288</td>
<td>20288</td>
<td>20288</td>
<td>20288</td>
<td>20288</td>
</tr>
</tbody>
</table>
Table 3: Relative size of formulations.

<table>
<thead>
<tr>
<th></th>
<th>nbils</th>
<th>LP_nvars</th>
<th>LP_ncons</th>
<th>LP_nnonzeros</th>
<th>NLP_nvars</th>
<th>NLP_ncons</th>
</tr>
</thead>
<tbody>
<tr>
<td>m_P</td>
<td>2.93</td>
<td>1.31</td>
<td>2.18</td>
<td>1.12</td>
<td>1.03</td>
<td>1.05</td>
</tr>
<tr>
<td>m_Q</td>
<td>1.67</td>
<td>1.10</td>
<td>1.59</td>
<td>1.67</td>
<td>1.11</td>
<td>1.78</td>
</tr>
<tr>
<td>m_PQ</td>
<td>1.50</td>
<td>1.08</td>
<td>1.69</td>
<td>1.95</td>
<td>1.09</td>
<td>2.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>m_Q</td>
<td>3.17</td>
<td>1.42</td>
<td>2.88</td>
<td>3.09</td>
<td>1.33</td>
<td>3.02</td>
</tr>
<tr>
<td>m_PQ</td>
<td></td>
<td>12.27</td>
<td>3.08</td>
<td>8.40</td>
<td>3.60</td>
<td>13.63</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>11.58</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12.55</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9.49</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2.47</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16.78</td>
</tr>
</tbody>
</table>
Results on the relaxed linear programs

Figure 2: Performance profile \((\text{LP\_obj, } F_{\text{all}}, I_{\text{all}})\)
Figure 3: Number of instances that could be solved within time_limit.
Figure 4: Performance profiles for \((\text{NLP\_time}, \mathcal{F}_{\text{PQ}}, I < (I_{\text{all}}, \mathcal{F}_{\text{PQ}}))\)
Figure 5: Performance profile \((\text{NLP\_gap}, \mathcal{F}_{\text{all}}, \mathcal{I}_{\geq}(\mathcal{I}_{\text{all}}, \mathcal{F}_{\text{all}}))\)
CONCLUSION
• There is a trade-off between disaggregating commodities (increasing the size of formulations) versus strengthening the LPs’ lower bounds and improving the NLPs’ solve times or gaps.
CONCLUSION

• There is a trade-off between disaggregating commodities (increasing the size of formulations) versus strengthening the LPs’ lower bounds and improving the NLPs’ solve times or gaps.

• The PQ-formulations outperform the Q-formulations.
CONCLUSION

- There is a trade-off between disaggregating commodities (increasing the size of formulations) versus strengthening the LPs’ lower bounds and improving the NLPs’ solve times or gaps.
- The PQ-formulations outperform the Q-formulations.
- MCF-J-PQ is slightly better than MCF-I-PQ.
CONCLUSION

- There is a trade-off between disaggregating commodities (increasing the size of formulations) versus strengthening the LPs’ lower bounds and improving the NLPs’ solve times or gaps.
- The PQ-formulations outperform the Q-formulations.
- MCF-J-PQ is slightly better than MCF-I-PQ.
- MCF-(I+J)-PQ and MCF-(I×J)-PQ only marginally strengthen the LPs’ lower bounds and often show some of the worst NLP solve times or gaps.
Figure 6: Size of the relaxed linear programs versus mean of performance ratio for $\text{LP}_{\text{obj}}$
Figure 7: Size of the relaxed linear programs versus mean of performance ratio for NLP_time
Figure 8: Size of the relaxed linear programs versus mean of performance ratio for NLP_gap
AN APPLICATION
The port of Newcastle, Australia, is the world’s largest coal export port with a throughput of **158.4 million tonnes** in 2014.

Coal is a blended product that is made-to-order according to customers’ desired product qualities.

Deviations from these target qualities result in contractually agreed **bonuses and penalties**.
**Figure 9:** Coal supply chain

**Figure 10:** Bonus/penalty function
(a) Given data

(b) Corresponding pooling problem

Figure 11: Example of given data and the corresponding pooling problem.
Acknowledgements: This research was supported by the ARC Linkage Grant no. LP110200524, Hunter Valley Coal Chain Coordinator (hvccc.com.au) and Triple Point Technology (tpt.com).
Questions?
M. Alfaki and D. Haugland.  
A multi-commodity flow formulation for the generalized pooling problem.  

M. Alfaki and D. Haugland.  
Strong formulations for the pooling problem.  

A. Ben-Tal, G. Eiger, and V. Gershovitz.  
Global minimization by reducing the duality gap.  
C. A. Haverly.
Studies of the behavior of recursion for the pooling problem.

G. P. McCormick.
Computability of global solutions to factorable nonconvex programs: Part I – convex underestimating problems.

H. D. Sherali and W. P. Adams.
A reformulation-linearization technique for solving discrete and continuous nonconvex problems, volume 31 of *Nonconvex Optimization and its Applications*.