Carnegie Mellon University

CAPD CENTER

Decomposition method for the Multiperiod Blending Problem

Francisco Trespalacios, Irene Lotero and Ignacio Grossmann

March 12, 2014

Center for Advanced Process Decision-making
Department of Chemical Engineering Carnegie Mellon University
Pittsburgh, PA 15213

Motivation and goals

Motivation

Multiperiod blending problem is a general model for many applications, and it is difficult to solve

- Gasoline and crude oil blending are some of the applications
- The model contains mixed-integer variables and bilinear constraints

Goals

Generate “good solutions” fast

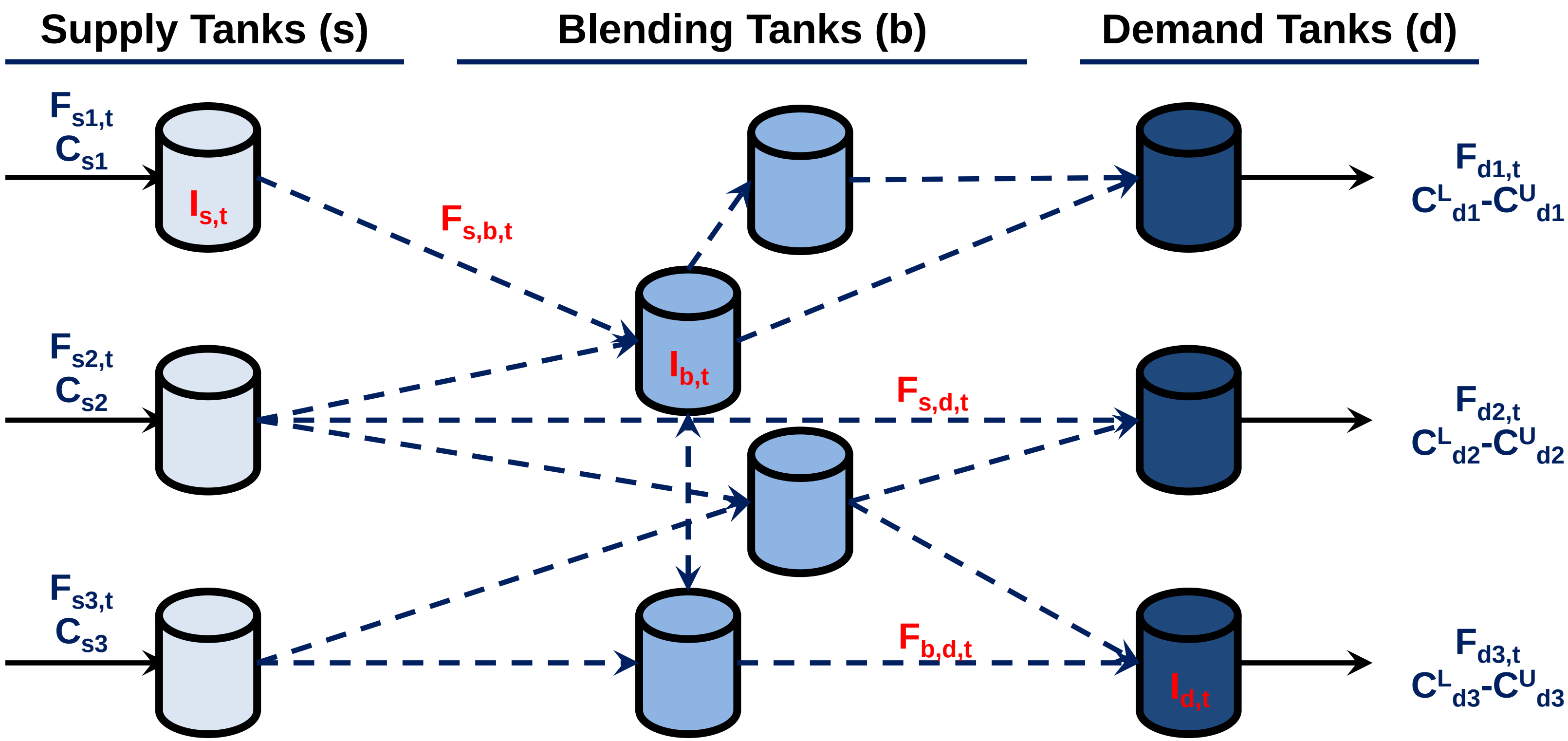
- Guaranteeing global optimality is not a priority
- Solutions must be feasible

Approach

Decompose the problem to simplify search for feasible solutions

- Solving smaller MINLPs with fewer 0-1 variables and bilinear terms
- “Guided” by an MILP relaxation of the problem

Multiperiod blending problem is defined over supply, blending and demand tanks



Given:
- Topology, initial conditions and flow profit/costs
- Supply tank flow and concentration
- Demand tank flow and concentration limits

Determine
- Flows between which tanks in which time periods
- Inventories/concentrations for tanks in each period
- **Maximum total profit of blending operation**

Assumptions:
- Supply concentrations are constant
- **No simultaneous input/output to blending tanks**
- Perfect mixing

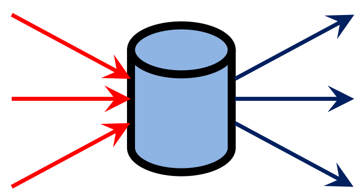
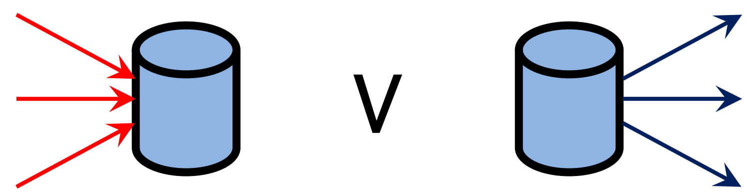
MINLP formulation contains bilinear terms

	Type
max profit - network flow cots	linear
s.t	
for flows into blending tanks: *[flow within bounds]* ⋁ *[flow = 0]*	mixed-integer linear
for flows into demand tanks: [*flow within bounds* / *concentrations within demand spec.*] ⋁ [*flow = 0* / *"no bounds" on concentration*]	mixed-integer linear
total inventory mass balance in tanks	linear
inventory mass balance by component in blending tanks	nonlinear (nonconvex)
no simultaneous in/out flow	integer linear
variable bounds	linear

Contains bilinear terms, for each contaminant, each blending tank and each time period

Possible to reduce bilinear terms using a disjunction

Requires 0-1 variable, but it does not increase combinatorial complexity (it might even reduce it!)[1]

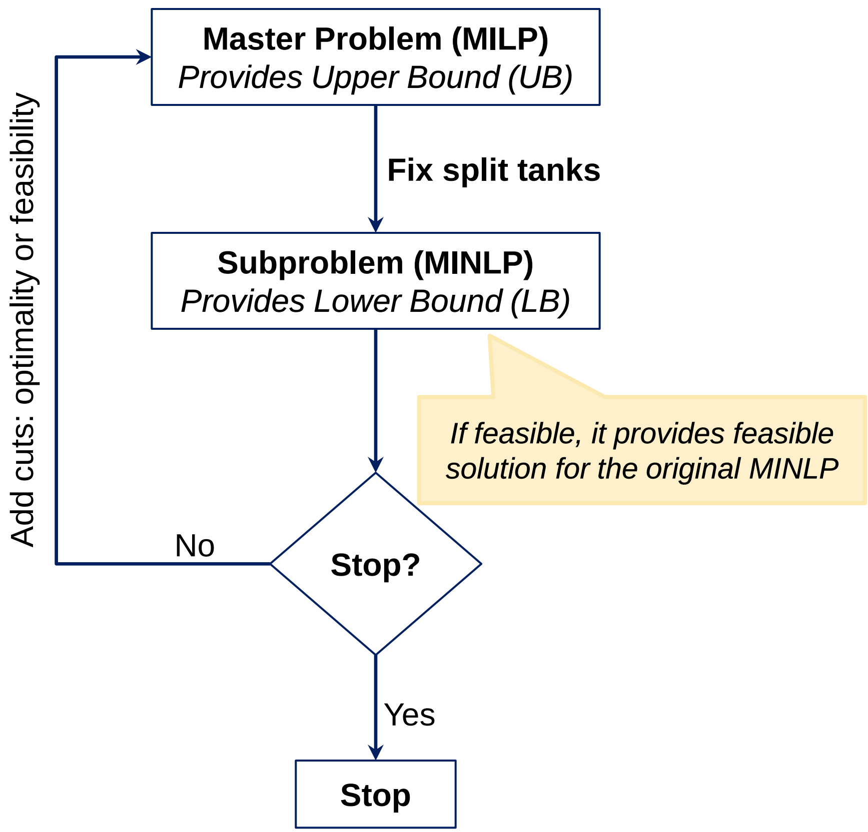
	Blend + Split tank	Blending tank	Split tank
Tank			
Mass balance	$I_{bt} = I_{bt-1} + \sum_{n \in S \cup B} F_{nbt} - \sum_{n \in B \cup D} F_{bnt}$	$I_{bt} = I_{bt-1} + \sum_{n \in S \cup B} F_{nbt}$	$I_{bt} = I_{bt-1} - \sum_{n \in B \cup D} F_{bnt}$
Comp. mass balance	$I_{bt}C_{qbt} = I_{bt-1}C_{qbt-1} + \sum_{s \in S} F_{sbt} C_{qs}^{IN} + \sum_{b' \in B} F_{b'bt} C_{qb't-1} - \sum_{n \in B \cup D} F_{bnt} C_{qbt-1}$	$I_{bt}C_{qbt} = I_{bt-1}C_{qbt-1} + \sum_{s \in S} F_{sbt} C_{qs}^{IN} + \sum_{b' \in B} F_{b'bt} C_{qb't-1}$	$C_{qbt} = C_{qbt-1}$

If some of the tanks are fixed as “split tanks”, the resulting MINLP becomes easier to solve

1. The new binary variable can in fact be continuous due to the problem structure. Furthermore, using it as binary variable and giving it priority over the network variables can increase the solution time of a branch-and-bound method

Decomposition algorithm seeks to find feasible solutions in short periods of time

Algorithm



Decisions in the algorithm

Stop criteria

- UB and LB gap
- Solution ime

Master MILP (MINLP relaxation)

- McCormick
- Piecewise McCormick
- Multiparametric disaggregation

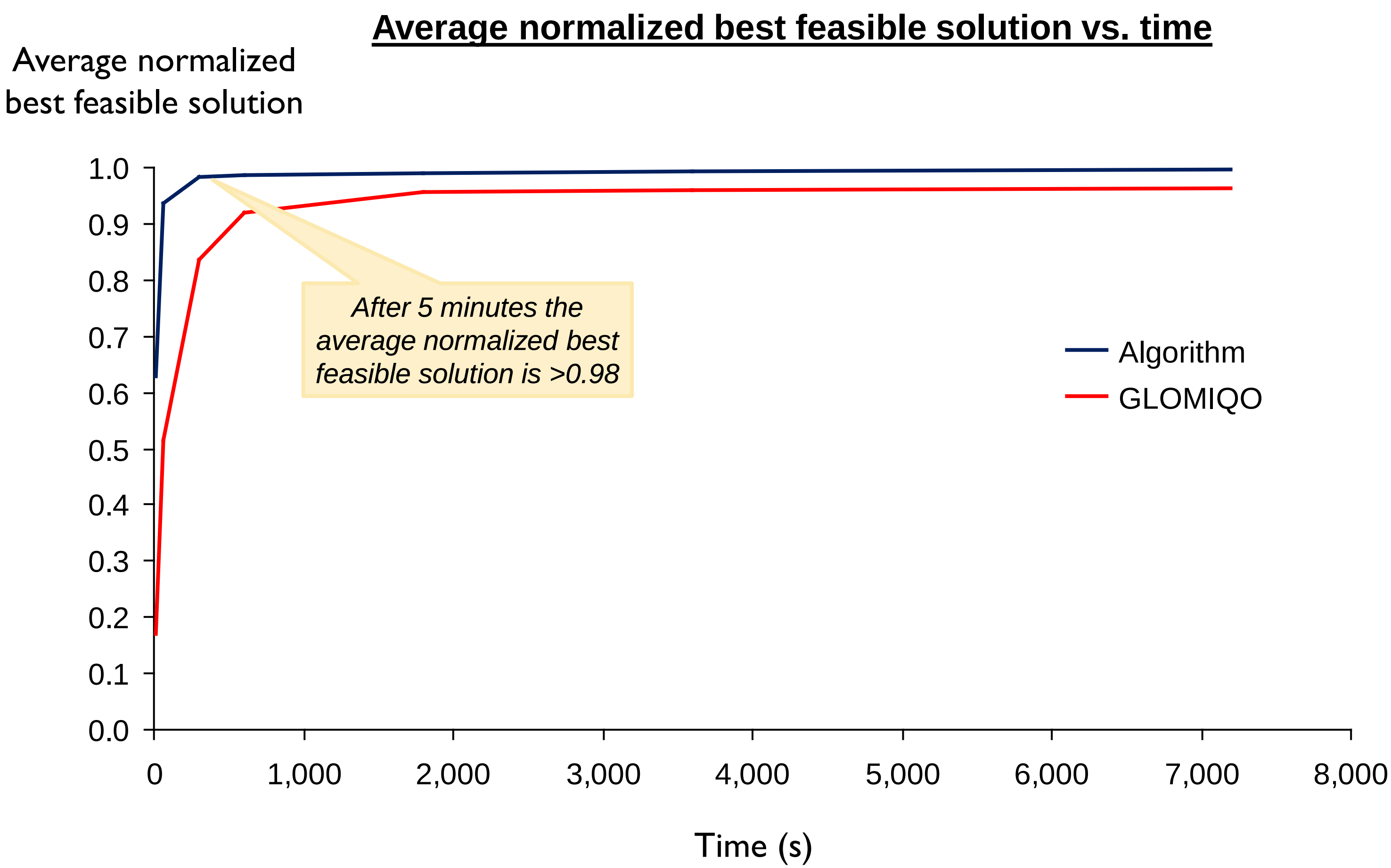
Solving MINLP

- Convex MINLP solver provides feasible solution, but optimality cut is not valid
- Global solution though commercial solver
- Global solution through specialized technique

Other considerations

- “no-good” are included in the subproblem, eliminating regions already evaluated in previous subproblems

Algorithm finds good feasible solutions fast (I/II)



Algorithm finds good feasible solutions fast (II/II)

Instance	Best know	Best feasible solution after 60 s: GLOMIQO	Best feasible solution after 60 s: Algorithm	Best feasible solution after 300 s: GLOMIQO	Best feasible solution after 300 s: Algorithm	Best feasible solution after 7200 s: GLOMIQO	Best feasible solution after 7200 s: Algorithm
3P-4Qa	45.2	40.6	43.8	40.6	45.2	44.4	45.2
3P-4Qb	13.5*	13.1	13.5	13.3	13.5	13.5	13.5
3P-8Qa	39.6	NA	32.5	NA	36.4	30.1	37.7
3P-8Qb	11.5*	NA	7.7	7.9	10.9	10.8	11.5
4P-2Qa	9.2*	9.2	9.2	9.2	9.2	9.2	9.2
4P-2Qb	54.0*	50.9	53.8	54.0	53.8	54.0	54.0
4P-4Q	20.0*	NA	19.7	16.9	20	19.9	20
4P-6Qa	9.2*	9.2	9.2	9.2	9.2	9.2	9.2
4P-6Qb	20.0*	7.4	19.4	19.3	20	20	20
4P-8Q	9.2	NA	9.2	9.2	9.2	9.2	9.2

2 prob with gap < 5%

5 problems with "bad" or no feas. sol.

1 problem with "bad" feas. sol.

8 problems with gap < 5% of best solution

3 problems with "bad" or no feas. sol.

Best known solution in 4 problems

Best known solution in 7 problems

Good solution in all problems

Best known solution in 6 problems

Best known solution in 9 problems

* = Optimal within 0.001 gap

Good MILP approximation and decomposition allow finding good solutions in first iteration

	Instance		First MINLP		First iteration (normalized[1])		
Instance	**0-1 vars**	**Bilinear terms**	**0-1 vars**	**Bilinear terms**	**UB**	**LB**	**Time (s)**
3P-4Qa	72	208	32	48	1.05	0.83	12
3P-4Qb	72	208		36	1.04	1.0	20
3P-8Qa	72	416	37	128	1.19	0.82	53
3P-8Qb	72	416	37	96	1.22	0.57	25
4P-2Qa	96	158	36	22	1.15	0.99	18
4P-2Qb	96	152	26	10	1.01	0.99	3
4P-4Q	96	292		64	1.07	0.97	8
4P-6Qa	96	474	36	66	1.15	0.99	27
4P-6Qb	96	438	41	96	1.07	0.97	6
4P-8Q	96	632	36	104	1.15	1.0	27

50%-70% fewer bilinear terms

93% fewer bilinear terms

50%-70% fewer 0-1 variables

~85% fewer bilinear terms

Good solution after 1st iteration

1st iteration in less than a minute in all problems

1. Normalized to best known solution

Test problems still small for industrial problem size

Industrial application problem size comparison

	Current examples	Gasoline blending	Crude blending
# of tanks	4	4 ✓	~45 ✗
# of qualities	2 - 8	>10 ✓	>10 ✓
Time periods	3 - 4	~20 ✗	~20 ✗

Future work to address applicability to industrial applications

Generate new instances that match industrial applications

- Increase problem sizes
- Adjust instance generation to resemble operating conditions

Improve solution time of subproblem

- Using MINLP alternative techniques
- Include a step in the algorithm to improve variable bounds

Address the complexity of increased problem sizes (time periods)