

Decomposition Methods

1.0 Introduction

Consider the XYZ corporation that has 10 departments, each of which have a certain function necessary to the overall productivity of the corporation. The corporation has capability to make 100 different products, but in any particular month, it makes some of them and does not make others. The decision of which products to make is made by the CEO. The CEO's decision is a yes/no decision on each of the 100 products. Of course, the CEO's decision depends, in part, on the productivity of the departments and their capability to make a profit given the decision of which products to make.

Each department knows its own particular business very well, and each has developed sophisticated mathematical programs (optimization problems) which provide the best (most profitable) way to use their resources given identification of which products to make.

The organization works like this. The CEO makes a tentative decision on which products to make, based on his/her own mathematical program which assumes certain profits from each department based on that decision. He/she then passes that decision to the various departments. Each of the departments uses that information to determine how it is going to operate in order to maximize profitability. Then each department passes that information back to the CEO. Some departments may not be able to be profitable at all with the CEO's selection of products to make: these departments will also communicate this information to the CEO.

Once the CEO gets all the information back from the departments, he or she will re-run their optimization to select the products, likely resulting in a modified choice of products, and then the process will

repeat. At some point, the optimization problem solved by the CEO will not change from one iteration to the next. At this point, the CEO will believe the current selection of products is best.

This is an example of a *multidivisional problem* [1, pg. 219]. Such problems involve coordinating the decisions of separate divisions, or departments, of a large organization, when the divisions operate autonomously. Solution of such problems often may be facilitated by separating them into a single master problem and subproblems where the master corresponds to the problem addressed by the CEO and the subproblems correspond to the problems addressed by the various departments.

We developed our example where the master problem involved choice of integer variables and the subproblems involved choice of continuous variables. The reason for this is that it conforms to the form of a mixed-integer-programming (MIP) problem, which is the kind of problem we have most recently had interest.

However, the master-subproblem relationship may be otherwise. It may also involve decisions on the part of the CEO to directly modify resources for each department. By “resources,” we mean the right-hand-side of the constraints. Such a scheme is referred to as a *resource-directed* approach.

Alternatively, the master-subproblem relationship may involve decisions on the part of the CEO to indirectly modify resources by charging each department a price for the amount of resources that are used. The CEO would then modify the prices, and the departments would adjust accordingly. Such a scheme is called a *price-directed* approach.

These types of optimization approaches are referred to as decomposition methods.

2.0 Connection with optimization: problem structure [2]

Recall that our optimization problems were always like this:

$$\begin{aligned} & \text{Minimize } f(\underline{x}) \\ & \text{Subject to } \underline{c}_1 \underline{x} \leq \underline{b}_1 \\ & \quad \underline{c}_2 \underline{x} \leq \underline{b}_2 \\ & \quad \dots \\ & \quad \underline{c}_m \underline{x} \leq \underline{b}_m \end{aligned}$$

We may place all of the row-vectors c_i into a matrix, and all elements b_i into a vector \underline{b} , so that our optimization problem is now:

$$\begin{aligned} & \text{Minimize } f(\underline{x}) \\ & \text{Subject to } \underline{A} \underline{x} \leq \underline{b} \end{aligned}$$

Problems that have special structures in the constraint matrices \underline{A} are typically more amenable to decomposition methods. Almost all of these structures involve the constraint matrix \underline{A} being block-angular. A block angular constraint matrix is illustrated in Fig. 1. In this matrix, the yellow-cross-hatched regions represent sub-matrices that contain non-zero elements. The remaining sub-matrices, not yellow-cross-hatched, contain all zeros. We may think of each yellow-cross-hatched as a department. The decision variables x_1 are important only to department 1; the decision variables x_2 are important only to department 2; and the decision variables x_3 are important only to department 3. In this particular structure, we have no need of a CEO at all. All departments are completely independent!

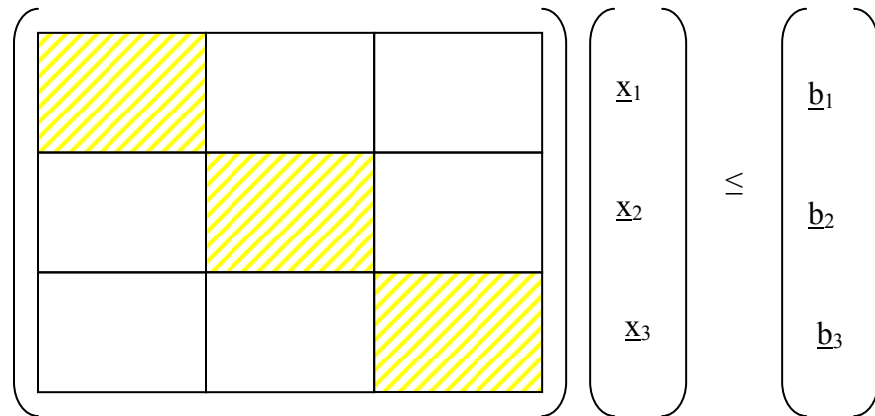


Fig. 1: Block-angular structure

Fig. 2 represents a structure where the decisions are linked at the CEO level, who must watch out for the entire organization's consumption of resources. In this case, the departments are independent, i.e., they are concerned only with decision on variables for which no other department is concerned, BUT... the CEO is concerned with constraints that span across the variables for all departments. And so we refer to this structure as block-angular with linking constraints.

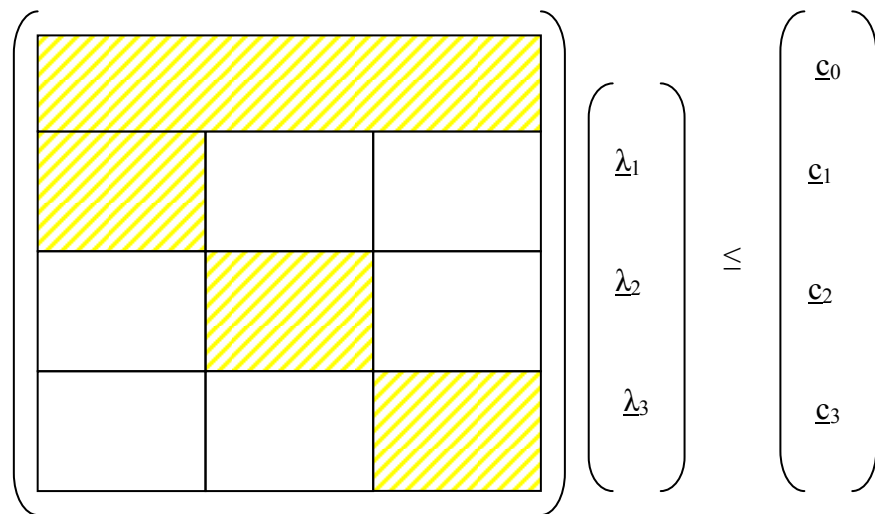


Fig. 2: Block-angular structure with linking constraints

This previous situation, characterized by Fig. 2, is not, however, the situation that we originally described. In the situation of Fig. 2, the CEO allocates resources.

In the original description, the CEO chose values (1 or 0) for certain variables for which it was assumed would affect each department. The original situation would have different departments linked by variables, then, and not by constraints. The structure of the constraint matrix for this situation is shown in Fig. 3.

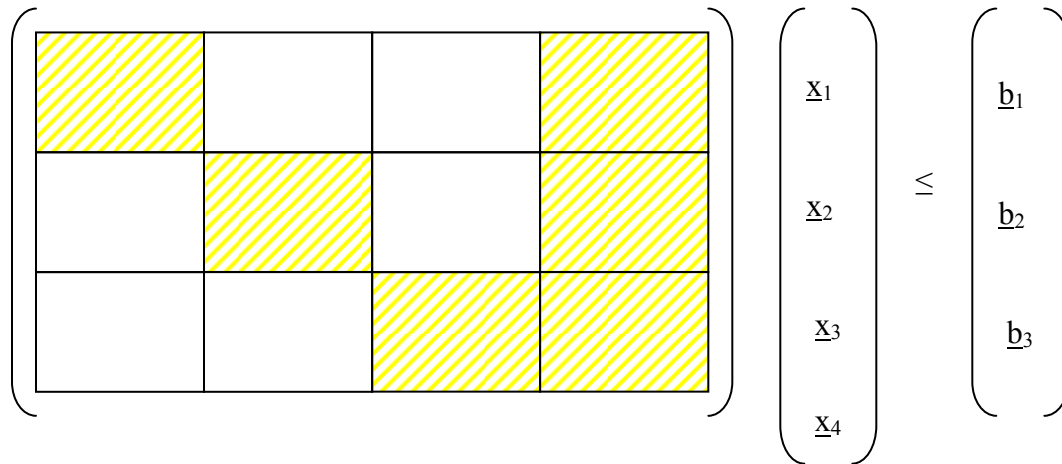


Fig. 3: Block-angular structure with linking variables

3.0 Motivation for decomposition methods: solution speed

To motivate decomposition methods, we consider introducing security constraints to a familiar problem: the OPF.

The OPF may be posed as problem P_0 .

$$\begin{array}{ll}
 \text{Min} & f_0(x_0, u_0) \\
 \text{s.t.} & h_k(x_k, u_0) = 0 \quad k = 0 \\
 & g_k(x_k, u_0) \leq g_k^{\max} \quad k = 0
 \end{array}$$

P_0

where $h_k(x_k, u_0) = 0$ represents the power flow equations and $g_k(x_k, u_0) \leq g_k^{\max}$ represents the line-flow constraints. The index $k=0$ indicates this problem is posed for only the “normal condition,” i.e., the condition with no contingencies.

Denote the number of constraints for this problem as N .

Assumption: Let's assume that running time T of the algorithm we use to solve the above problem is proportional to the square of the number of constraints, i.e., N^2 . For simplicity, we assume the constant of proportionality is 1, so that $T = N^2$.

Now let's consider the security-constrained OPF (SCOPF). Its problem statement is given as problem P_c :

$$\begin{aligned}
 P_c \quad & \text{Min} \quad f_0(x_0, u_0) \\
 & \text{s.t.} \quad h_k(x_k, u_0) = 0 \quad k = 0, 1, 2, \dots, c \\
 & \quad \quad g_k(x_k, u_0) \leq g_k^{\max} \quad k = 0, 1, 2, \dots, c
 \end{aligned}$$

Notice that there are c contingencies to be addressed in the SCOPF, and that there are a complete new set of constraints for each of these c contingencies. Each set of contingency-related constraints is exactly like the original set of constraints (those for problem P_0), except it corresponds to the system with an element removed.

So the SCOPF must deal with the original N constraints, and also another set of N constraints for every contingency. Therefore, the total number of constraints for Problem P_c is $N + cN = (c+1)N$.

Under our assumption that running time is proportional to the square of the number of constraints, then the running time will be proportional to $[(c+1)N]^2 = (c+1)^2 N^2 = (c+1)^2 T$.

What does this mean?

It means that the running time of the SCOPF is $(c+1)^2$ times the running time of the OPF. So if it takes OPF 1 minute to run, and you want to run SCOPF with 100 contingencies, it will take you 101^2 minutes, or 10,201 minutes to run the SCOPF. This is 170 hours, about 1 week!!!!

Many systems need to address 1000 contingencies. This would take about 2 years!

So this is what you do.....

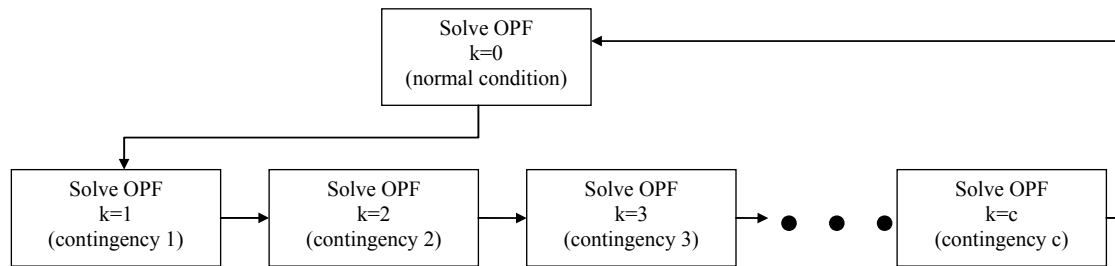


Fig. 4: Decomposition solution strategy

The solution strategy first solves the OPF (master problem) and then takes contingency 1 and re-solves the OPF, then contingency 2 and resolves the OPF, and so on (these are the subproblems). For any contingency-OPFs which require a redispatch, relative to the $k=0$ OPF, an appropriate constraint is generated, and at the end of the cycle these constraints are gathered and applied to the $k=0$ OPF. Then the $k=0$ OPF is resolved, and the cycle starts again. Experience has it that such an approach usually requires only 2-3 cycles.

Denote the number of cycles as m .

Each of the individual problems has only N constraints and therefore requires only T minutes.

There are $(c+1)$ individual problems for every cycle.

There are m cycles.

So the amount of running time is $m(c+1)T$.

If $c=100$ and $m=3$, $T=1$ minute, this approach requires 303 minutes. That would be about 5 hours (instead of 1 week).

If $c=1000$ and $m=3$, $T=1$ minute, this approach requires about 50 hours (instead of 2 years).

In addition, this approach is easily parallelizable, i.e., each individual OPF problem can be sent to its own CPU. This will save even more time.

Fig. 5 compares computing time for a “toy” system. The comparison is between a full SCOPF, a decomposed SCOPF (DSCOPF), and a decomposed SCOPF where the individual OPF problems have been sent to separate CPUs.

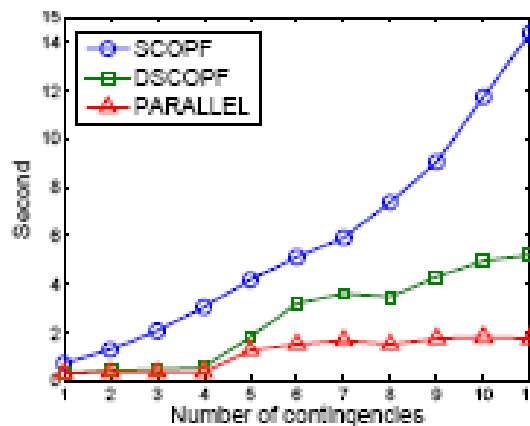


Fig. 5

4.0 Benders decomposition

J. F. Benders [3] proposed solving a mixed-integer programming problem by partitioning the problem into two parts – an integer part and a continuous part. It uses the branch-and-bound method on the integer part and linear programming on the continuous part.

The approach is well-characterized by the linking-variable problem illustrated in Fig. 3 where here the linking variables are the integer variables. In the words of A. Geoffrion [4], “J.F. Benders devised a clever approach for exploiting the structure of mathematical programming problems with complicating variables (variables which, when temporarily fixed, render the remaining optimization problem considerably more tractable).”

Note in the below problem statements, all variables except z_1^* and z_2^* are considered to be vectors.

There are three steps to Benders decomposition for mixed integer programs.

The problem can be generally specified as follows:

$$\begin{aligned} \max z_1 &= c_1^T x + c_2^T w \\ \text{s.t.} \\ Dw &\leq e \\ A_1 x + A_2 w &\leq b \\ x, w &\geq 0 \\ w &\text{ integer} \end{aligned}$$

Define the master problem and primal subproblem as

Master :

$$\begin{aligned} \max z_1 &= c_2^T w + z_2^* \\ \text{s.t.} \\ Dw &\leq e \\ w &\geq 0 \\ w &\text{ integer} \end{aligned}$$

Primal subproblem :

$$\begin{aligned} \max z_2 &= c_1^T x \\ \text{s.t.} \\ A_1 x &\leq b - A_2 w^* \\ x &\geq 0 \end{aligned}$$

Some comments on duality for linear programs:

1. Number of dual decision variables is number of primal constraints.
Number of dual constraints is number of primal decision variables.
2. Coefficients of decision variables in dual objective are right-hand-sides of primal constraints.

<p><u>Problem P</u></p> $\max F = 3x_1 + 5x_2$ <p>s.t. $x_1 \leq 4$</p> $2x_2 \leq 12$ $3x_1 + 2x_2 \leq 18$ $x_1 \geq 0, x_2 \geq 0$ <p style="text-align: center;">Primal Problem</p>	\longleftrightarrow	<p><u>Problem D</u></p> $\min G = 4\lambda_1 + 12\lambda_2 + 18\lambda_3$ <p>subject to</p> $\lambda_1 + 3\lambda_3 \geq 3$ $2\lambda_2 + 2\lambda_3 \geq 5$ $\lambda_1 \geq 0, \lambda_2 \geq 0, \lambda_3 \geq 0$ <p style="text-align: center;">Dual Problem</p>
--	-----------------------	---

3. Coefficients of decision variables in primal objective are right-hand-sides of dual constraints.

<p><u>Problem P</u></p> $\max F = 3x_1 + 5x_2$ <p>s.t. $x_1 \leq 4$</p> $2x_2 \leq 12$ $3x_1 + 2x_2 \leq 18$ $x_1 \geq 0, x_2 \geq 0$ <p style="text-align: center;">Primal Problem</p>	\longleftrightarrow	<p><u>Problem D</u></p> $\min G = 4\lambda_1 + 12\lambda_2 + 18\lambda_3$ <p>subject to</p> $\lambda_1 + 3\lambda_3 \geq 3$ $2\lambda_2 + 2\lambda_3 \geq 5$ $\lambda_1 \geq 0, \lambda_2 \geq 0, \lambda_3 \geq 0$ <p style="text-align: center;">Dual Problem</p>
--	-----------------------	---

4. Coefficients of one variable across multiple primal constraints are coefficients of multiple variables in one dual constraint.

<p><u>Problem P</u></p> $\max F = 3x_1 + 5x_2$ <p>s.t. $x_1 \leq 4$</p> $2x_2 \leq 12$ $3x_1 + 2x_2 \leq 18$ $x_1 \geq 0, x_2 \geq 0$ <p style="text-align: center;">Primal Problem</p>	\longleftrightarrow	<p><u>Problem D</u></p> $\min G = 4\lambda_1 + 12\lambda_2 + 18\lambda_3$ <p>subject to</p> $\lambda_1 + 3\lambda_3 \geq 3$ $2\lambda_2 + 2\lambda_3 \geq 5$ $\lambda_1 \geq 0, \lambda_2 \geq 0, \lambda_3 \geq 0$ <p style="text-align: center;">Dual Problem</p>
--	-----------------------	---

5. If primal obj. is maximization, dual obj. is minimization.
6. If primal constraints are \leq , dual constraints are \geq .

From the above, we can write the dual of our primal subproblem.

Primal subproblem :

$$\max z_2 = c_1^T x$$

s.t.

$$A_1 x \leq b - A_2 w^*$$

$$x \geq 0$$

Dual subproblem :

$$\min z_2 = (b - A_2 w^*)^T \lambda$$

s.t.

$$A_1^T \lambda \geq c_1$$

$$\lambda_k \geq 0, \quad \forall k$$



Now consider the master problem and dual subproblem together:

Master :

$$\max z_1 = c_2^T w + z_2^*$$

s.t.

$$Dw \leq e$$

$$w \geq 0$$

w integer

Dual subproblem :

$$\min z_2 = (b - A_2 w^*)^T \lambda$$

s.t.

$$A_1^T \lambda \geq c_1$$

$$\lambda_k \geq 0, \quad \forall k$$

1. *Interdependence:* The master depends on the subproblem optimal objective z_2^* , and the subproblem depends on the master optimal solution w^* . Therefore, the solution to each problem depends on the solution obtained in the other problem.
2. *Iterative procedure:* We will solve the overall problem by iterating between the master and the subproblem. The master will be used to generate a solution w^* , given a value (or a guess) for z_2^* . Then the subproblem will be used to get a new value of z_2^* and λ^* using the solution w^* obtained in the master. This will tell us one very important thing: when we resolve the master, we should constrain z_2^* to be no bigger than $(b - A_2 w^*)^T \lambda^*$, i.e., $z_2^* \leq (b - A_2 w^*)^T \lambda^*$.

3. *Upper bound:*

- a. *Initial solution:* Start the solution procedure by solving the master problem with a guess for z_2^* . Since the dual problem is going to minimize (lower) z_2 , let's be safe and guess a large value of z_2^* at this initial master problem solution. Since this value of z_2^* is chosen large, we can be sure that the solution to the master, z_1^* , will be above the actual (overall problem optimal) solution, and so we will consider this solution z_1^* to be an upper bound on the actual solution.
- b. *Successive solutions:* As the iterations proceed, we will add constraints, so that the master problem solution z_1^* , will continuously decrease to either the actual (overall problem optimal) solution, or it will decrease to value greater than the actual solution.

Thus, the value of z_1^* , obtained from the master problem, serves as an upper bound on the actual (overall problem optimal) solution.

4. *Lower bound:* The dual problem results in a new value of z_2^* , and it can then be added to $c_2^T w^*$ (where w^* was obtained from the last master problem solution) to provide another estimate of z_1^* . Since the dual problem minimizes z_2 , the solution $c_2^T w^* + z_2^*$ will be a lower bound on z_1 . These constraints are referred to as optimality cuts.
5. *Monotonicity:* The upper bound decreases monotonically through the iterations, but the lower bound may not.
6. *Feasibility:* If the dual problem results in an unbounded solution, then it means the primal problem is infeasible. In this case, we must resolve the master problem with more restrictive constraints on w . The associated constraints are called feasibility cuts.
7. *Algorithm:* In what follows, we specify Q as the set of constraints binding the master program. It will change via the addition of feasibility and optimality cuts as the algorithm proceeds. Initially, $Q = \{w_k \leq \text{large number, for all } k, z_2^* \leq M, M \text{ large}\}$.

Master problem:

$$\begin{aligned} \max z_1 &= c_2^T w + z_2^* \\ \text{s.t.} \\ w_k &\text{ constrained as in } Q \quad \forall k \\ z_2^* &\leq M, \quad M \text{ is large} \\ w_k &\text{ integer } \quad \forall k \end{aligned}$$

Sub-problem (dual):

$$\begin{aligned} \min z_2 &= (b - A_2 w^*)^T \lambda \\ \text{s.t.} \\ A_1^T \lambda &\geq c_1 \\ \lambda_k &\geq 0, \quad \forall k \end{aligned}$$

1. Solve the master problem using Branch and Bound (or any other integer programming method). Designate the solution as w^* .
2. Using the value of w^* found in step 1, solve the sub-problem (the dual) which gives z_2^* and λ^* . There are two possibilities:
 - a. If the solution is unbounded (implying the primal is infeasible), adjoin the most constraining feasibility constraint from $(b - A_2 w)^T \lambda \geq 0$ to Q , and go to step 1.
 - b. Otherwise, designate the solution as λ^* and go to step 3.
3. Compare z_1 found in step 1 to $c_2^T w^* + z_2^*$ where $z_2^* = (b - A_2 w^*)^T \lambda^*$ found in step 2. There are two possibilities:
 - a. If they are equal (or within ε of each other), then the solution (w^*, λ^*) corresponding to the subproblem dual solution, is optimal and the primal variables x^* are found as the dual variables¹ within the subproblem.
 - b. If they are not equal, adjoin an optimality constraint to Q given by $z_2^* \leq (b - A_2 w^*)^T \lambda^*$ and go to step 1.

¹ (Dual variables are the coefficients of the objective function in the final iteration of the simplex method and are provided with the LP solution by a solver like CPLEX. They may be found from the fact that primal and dual objective functions must be equal at the optimum: $c^T x = (b - A_2 w^*)^T \lambda^*$.)

Step 3 is a check on Benders optimal rule.

Benders optimal rule: If (z_1^*, w^*) is the optimal solution to the master problem, and (z_2^*, λ^*) is the optimal solution to dual subproblem, and if

$$\underbrace{c_2^T w^*}_{\text{from master problem}} + \underbrace{(b - A_2 w^*)^T \lambda^*}_{z_2^* \text{ from subproblem}} = \underbrace{z_1^*}_{\text{from master problem}}$$

then (z_1^*, w^*, λ^*) is the optimal solution for the complete problem.

We will work an example using the formalized nomenclature of the previous summarized steps.

Consider the following problem P_0 [5]:

$$\begin{array}{ll}
 \max & z_1 = 4x_1 + 3x_2 + 5w \\
 \text{subject to :} & 2x_1 + 3x_2 + w \leq 12 \\
 & 2x_1 + x_2 + 3w \leq 12 \\
 & w \leq 20 \\
 & x_1, x_2, w \geq 0, \\
 & w \text{ integer}
 \end{array}$$

Clearly P_0 is a mixed integer problem.

There are two ways to think about this problem.

First way: Let's redefine the objective function as

$$z_1 = 5w + z_2$$

where

$$z_2 = 4x_1 + 3x_2$$

so that problem P_{1P} below is equivalent to problem P_0 above:

$$\begin{array}{l}
P_{1P} \\
\max \quad z_1 = 5w + \left(\begin{array}{l} \max \quad z_2 = 4x_1 + 3x_2 \\ \text{subject to} \quad 2x_1 + 3x_2 \leq 12 - w \\ \quad \quad \quad 2x_1 + x_2 \leq 12 - 3w \\ \quad \quad \quad x_1, x_2 \geq 0 \end{array} \right) \\
\text{subject to:} \quad w \leq 20 \\
\quad \quad \quad w \geq 0, \\
\quad \quad \quad w \text{ integer}
\end{array}$$

This way is similar to the way J. Bloom described his two-stage generation planning problem in [6].

Second way: Here, we will simply cast the problem into the general form outlined in our three-step procedure.

Comparing to our general formulation,

$$\begin{array}{l}
\max z_1 = c_1^T x + c_2^T w \\
s.t. \\
A_1 x + A_2 w \leq b \\
x, w \geq 0 \\
w \text{ integer}
\end{array}$$

we find that

$$c_1 = \begin{bmatrix} 4 \\ 3 \end{bmatrix}, \quad c_2 = 5 \\
A_1 = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \quad b = \begin{bmatrix} 12 \\ 12 \end{bmatrix}$$

Step 1: The master problem is

$$\max z_1 = c_2^T w + z_2^*$$

s.t.

$$Q: w_k \text{ constrained } \forall k, z_2^* \leq M, M \text{ is large}$$

$$w_k \geq 0 \text{ and integer } \forall k$$

or

$$\max z_1 = c_2^T w + z_2^* \quad \max z_1 = 5w + z_2^*$$

s.t.

s.t.

$$Q: w \leq 20, z_2^* \leq M \rightarrow Q: w \leq 20, z_2^* \leq M$$

$$w \geq 0 \text{ and integer} \quad w \geq 0 \text{ and integer}$$

The solution to this problem is trivial: since the objective function is being maximized, we make w and z_2^* as large as possible, resulting in $w^*=20, z_2^*=M$.

Step 2: Using the value of w found in the master, get the dual:

$$\min z_2 = (b - A_2 w^*)^T \lambda \quad \min z_2 = \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w^* \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}$$

s.t.

s.t.

$$A_1^T \lambda \geq c_1$$

$$\rightarrow \begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$\lambda \geq 0$$

$$\lambda_1, \lambda_2 \geq 0$$

Substituting, from step 1, $w^*=20$, the subproblem becomes:

$$\min z_2 = \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} 20 \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = -8\lambda_1 - 48\lambda_2$$

s.t.

$$\begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$\lambda_1, \lambda_2 \geq 0$$

Because the λ_k 's are non-negative, all terms in the objective function are negative. Noting the λ_k 's are constrained from below, we may make them as large as we like, implying the objective function is unbounded. This occurs because the coefficients in the objective function are negative. The coefficients in the objective function are negative because the master problem, since it was not sufficiently constrained, yielded a poor choice of w . We need to correct this situation, by taking step 2b, which means we will add a “feasibility constraint” to the master problem. This feasibility constraint is contained in $(b-A_2w)^T \lambda \geq 0$, or

$$\left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq 0$$

or

$$(12 - w)\lambda_1 + (12 - 3w)\lambda_2 \geq 0$$

To guarantee the above, without concern for what values of λ_k are chosen, we must make

$$(12 - w) \geq 0, \quad (12 - 3w) \geq 0$$

resulting in

$$12 \geq w, \quad 4 \geq w$$

Clearly, w must be chosen to satisfy $w \leq 4$. This constraint is added to Q , and we repeat step 1.

Step 1:

$$\max z_1 = 5w + z_2^*$$

s.t.

$$Q: w \leq 20, \quad w \leq 4, \quad z_2^* \leq M$$

$$w \geq 0 \quad \text{and integer}$$

The solution is clearly $w=4, z_2^*=M$, with $z_1^*=5(4)+M=20+M$.

Step 2: Using the value of $w=4$ found in the master, get the dual.

$$\min z_2 = (b - A_2 w^*)^T \lambda \qquad \min z_2 = \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w^* \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}$$

s.t.

s.t.

$$A_1^T \lambda \geq c_1$$

$$\rightarrow \begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$\lambda \geq 0$$

$$\lambda_1, \lambda_2 \geq 0$$

Substituting, from step 1, $w^*=4$, the subproblem becomes:

$$\min z_2 = \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} 4 \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 8\lambda_1$$

s.t.

$$\begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$\lambda_1, \lambda_2 \geq 0$$

We can use CPLEX LP solver (or any other LP solver) to solve the above, obtaining the solution $\lambda_1^*=0, \lambda_2^*=3$, with objective function value $z_2^*=0$. (Intuitively, one observes that minimization of the objective subject to nonnegativity constraint on λ_1 requires $\lambda_1=0$; then λ_2 can be anything as long as it satisfies

$$2\lambda_2 \geq 4 \Rightarrow \lambda_2 \geq 2$$

$$\lambda_2 \geq 3$$

Therefore a finite optimal solution is $\lambda_1^*=0, \lambda_2^*=3$.)

Since we have a bounded dual solution, our primal is feasible, and we may proceed to step 3.

Step 3: Compare z_1 found in step 1 to $c_2^T w^* + z_2^*$ where $z_2^* = (b - A_2 w^*)^T \lambda^*$ is found in step 2.

In step 1, solution of the master problem resulted in $z_1^* = 20 + M$.

In step 2, solution of the subproblem resulted in $z_2^* = 0$.

In both problems, $c_2 = 5$, and we found (master) or used (sub) $w^* = 4$.

Benders optimal rule is

$$\underbrace{c_2^T w^*}_{\text{from master problem}} + \underbrace{(b - A_2 w^*)^T \lambda^*}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{z_1^*}_{\text{from master problem}}$$

Substitution yields:

$$\underbrace{5 \bullet 4}_{\text{from master problem}} + \underbrace{0}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{20 + M}_{\text{from master problem}}$$

The fact that they are not equal indicates that our solution is not optimal, since it does not satisfy Benders optimal rule. These two problems, the master and the subproblem, are really part of a single problem, and therefore for the single problem to be solved, the solutions to the master and subproblems must be *consistent*. That is, when we maximize $z_1 = c_2^T w^* + z_2^*$ in the master using a value of z_2^* , we need to find this value to be the same as the solution that the subproblem gives for z_2^* . If we do that (since $c_2 w^*$ is the same for both), the objective function from the master problem, z_1^* , will be the same as the sum of $\{c_2^T w + z_2^*\}$ where z_2^* is the objective function from the subproblem.

If we find that z_2^* differs in the master and subproblem, as we have found here, then we impose a constraint in the master based on the answer obtained in the subproblem. The fact that this constraint is imposed in order to obtain optimality makes it an optimality constraint, or in the language of Benders, an *optimality cut*.

We distinguish between a feasibility cut and an optimality cut:

- *Feasibility cut*: Takes place as a result of finding an unbounded dual subproblem, which, by duality, implies an infeasible primal subproblem. It means that for the value of w found in the master problem, there is no possible solution in the primal subproblem. We address this by adding a feasibility cut (a constraint on w) to the master problem, where that cut is obtained from

$$(b - A_2 w)^T \lambda \geq 0.$$

- *Optimality cut*: Takes place as a result of finding that Benders optimal rule is not satisfied, i.e., that

$$\underbrace{c_2^T w^*}_{\text{from master problem}} + \underbrace{(b - A_2 w^*)^T \lambda^*}_{z_2^* \text{ from subproblem}} < \underbrace{z_1^*}_{\text{from master problem}}$$

It means that the value of z_2^* used in the master problem is larger than the value of z_2^* computed in the subproblem. We address this by adding an optimality cut (a constraint on z_2^*) to the master problem, where that cut is obtained from

$$z_2^* \leq (b - A_2 w^*)^T \lambda^*$$

Recalling that we are still in Step 3 of our example, and that we found that Benders optimal rule is not satisfied, we need to obtain the optimality cut from $z_2^* \leq (b - A_2 w^*)^T \lambda^*$. With

$$b = \begin{bmatrix} 12 \\ 12 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \quad \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 3 \end{bmatrix}$$

$$z_2^* \leq \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w \right)^T \begin{bmatrix} 0 \\ 3 \end{bmatrix} = [12 - w \quad 12 - 3w] \begin{bmatrix} 0 \\ 3 \end{bmatrix} = 36 - 9w$$

Now we return to step 1.

Step 1: Adjoin the optimality cut to Q , resulting in the following master problem:

$$\begin{aligned} \max \quad & z_1 = 5w + (z_2^*) \\ \text{subject to: } \quad & Q: w \leq 20, w \leq 4, z_2^* \leq M, z_2^* \leq 36 - 9w \\ & w \geq 0, w \text{ integer} \end{aligned}$$

This all-integer program can be solved using a branch and bound algorithm (both CPLEX and Matlab have one), but the solution can be identified using enumeration, since w can only be 0, 1, 2, 3, or 4. For example, letting $w=0$, we have

$$\begin{aligned} \max \quad & z_1 = (z_2^*) \\ \text{subject to: } \quad & Q: z_2^* \leq M, z_2^* \leq 36 \end{aligned}$$

The solution is recognized immediately, as $z_2^*=36, z_1^*=36$.

Likewise, letting $w=1$, we have

$$\begin{aligned} \max \quad & z_1 = 5 + (z_2^*) \\ \text{subject to: } \quad & Q: z_2^* \leq M, z_2^* \leq 27 \end{aligned}$$

The solution is recognized immediately, as $z_2^*=27, z_1^*=32$.

Continuing on, we find the complete set of solutions are

$$\begin{aligned} w=0, z_2^*=36, z_1=36 \\ w=2, z_2^*=27, z_1=32 \\ w=2, z_2^*=18, z_1=28 \\ w=3, z_2^*=9, z_1=24 \\ w=4, z_2^*=0, z_1=20 \end{aligned}$$

Since the first one results in maximizing z_1 , our solution is $w^*=0, z_2^*=36, z_1^*=36$.

Step 2: Using the value of w found in the master, get the dual:

$$\begin{aligned} \min z_2 &= (b - A_2 w^*)^T \lambda & \min z_2 &= \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w^* \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \\ \text{s.t.} & & \text{s.t.} & \\ A_1^T \lambda &\geq c_1 & \rightarrow & \begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix} \\ \lambda &\geq 0 & & \lambda_1, \lambda_2 \geq 0 \end{aligned}$$

Substituting, from step 1, $w^*=0$, the subproblem becomes:

$$\begin{aligned} \min z_2 &= \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} 0 \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 12\lambda_1 + 12\lambda_2 \\ \text{s.t.} & \\ \begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} &\geq \begin{bmatrix} 4 \\ 3 \end{bmatrix} \\ \lambda_1, \lambda_2 &\geq 0 \end{aligned}$$

We can use CPLEX LP solver (or any other LP solver) to solve the above, obtaining the solution $\lambda_1^*=2$, $\lambda_2^*=0$, with objective function value $z_2^*=24$. Since we have a bounded dual solution, our primal is feasible, and we may proceed to step 3.

Brief tutorial for using CPLEX.

CPLEX version 10.1.0 resides on an ISU server called pluto. To access it, you will need to logon to pluto. To do that, you will need a telnet and ftp facility. You may find instructions on getting the appropriate telnet and ftp facilities at http://home.eng.iastate.edu/~jdm/ee458/Intro_CPLEX.pdf.

The first thing to do is to construct a file containing the problem. To construct this file, you can use the program called “notepad” under the “accessories” selection of the start button in Windows.

Once you open notepad, you can immediately save to your local directory under the filename “filename.lp.” You can choose

“filename” to be whatever you want, but you will need the extension “lp.”

To obtain the extension “lp” when you save, you should do “save as” and then choose “all files.” Otherwise, it will assign the suffix “.txt” to your file.

Here is what I typed into the file I called “mip.lp”...

```
maximize
  12 x1 + 12 x2
subject to
  2 x1 + 2 x2 >= 4
  3 x1 +   x2 >= 3
  x1 >= 0
  x2 >= 0
end
```

Once you get the above file onto Pluto, may simply type
cplex101
CPLEX> read mip.lp
CPLEX> primopt

Detailed information on using CPLEX is available at http://home.eng.iastate.edu/~jdm/ee458/Intro_CPLEX.pdf.

Step 3: Compare z_1 found in step 1 to $c_2^T w^* + z_2^*$ where $z_2^* = (b - A_2 w^*)^T \lambda^*$ is found in step 2.

In step 1, solution of the master problem resulted in $z_1^* = 36$

In step 2, solution of the subproblem resulted in $z_2^* = 24$.

In both problems, $c_2 = 5$, and we found (master) or used (sub) $w^* = 0$.

Benders optimal rule is

$$\underbrace{c_2^T w^*}_{\text{from master problem}} + \underbrace{(b - A_2 w^*)^T \lambda^*}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{z_1^*}_{\text{from master problem}}$$

Substitution yields:

$$\underbrace{5 \bullet 0}_{\text{from master problem}} + \underbrace{24}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{36}_{\text{from master problem}}$$

Benders optimal rule is not satisfied, we need to obtain the optimality cut from $z_2^* \leq (b - A_2 w^*)^T \lambda^*$. With

$$b = \begin{bmatrix} 12 \\ 12 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \quad \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

$$z_2^* \leq \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w \right)^T \begin{bmatrix} 2 \\ 0 \end{bmatrix} = [12 - w \quad 12 - 3w] \begin{bmatrix} 2 \\ 0 \end{bmatrix} = 24 - 2w$$

Now we return to step 1.

Step 1: Adjoin the optimality cut to Q , resulting in the following master problem:

$$\max \quad z_1 = 5w + (z_2^*)$$

$$\text{subject to: } Q: \quad w \leq 20, \quad w \leq 4, \quad z_2^* \leq M, \quad z_2^* \leq 36 - 9w, \quad z_2^* \leq 24 - 2w \\ w \geq 0, \quad w \text{ integer}$$

This all-integer program can be solved using a branch and bound algorithm (both CPLEX and Matlab have one), but the solution can be identified using enumeration, since w can only be 0, 1, 2, 3, or 4.

For example, letting $w=0$, we have

$$\max \quad z_1 = (z_2^*)$$

$$\text{subject to: } Q: \quad z_2^* \leq M, \quad z_2^* \leq 36, \quad z_2^* \leq 24$$

The solution is recognized immediately, as $z_2^*=24, z_1^*=24$.

Likewise, letting $w=1$, we have

$$\begin{aligned} \max \quad & z_1 = 5 + (z_2^*) \\ \text{subject to: } \quad & Q: \quad z_2^* \leq M, z_2^* \leq 27, z_2^* \leq 22 \end{aligned}$$

The solution is recognized immediately, as $z_2^*=22, z_1^*=27$.

Continuing on, we find the complete set of solutions are

$$\begin{aligned} w=0, z_2^*=24, z_1=24 \\ w=1, z_2^*=22, z_1=27 \\ w=2, z_2^*=18, z_1=28 \\ w=3, z_2^*=9, z_1=24 \\ w=4, z_2^*=0, z_1=20 \end{aligned}$$

And so the third one results in maximizing z_1 , so our solution is $w^*=2, z_2^*=18, z_1^*=28$.

Step 2: Using the value of w found in the master, get the dual:

$$\begin{aligned} \min z_2 &= (b - A_2 w^*)^T \lambda & \min z_2 &= \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w^* \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \\ \text{s.t.} & & \text{s.t.} & \\ A_1^T \lambda &\geq c_1 & \rightarrow & \begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix} \\ \lambda &\geq 0 & & \lambda_1, \lambda_2 \geq 0 \end{aligned}$$

Substituting, from step 1, $w^*=2$, the subproblem becomes:

$$\begin{aligned} \min z_2 &= \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} 2 \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 10\lambda_1 + 6\lambda_2 \\ \text{s.t.} & \\ \begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} &\geq \begin{bmatrix} 4 \\ 3 \end{bmatrix} \\ \lambda_1, \lambda_2 &\geq 0 \end{aligned}$$

We can use CPLEX LP solver (or any other LP solver) to solve the above, obtaining the solution $\lambda_1^*=0.5, \lambda_2^*=1.5$, with objective

function value $z_2^*=14$. Since we have a bounded dual solution, our primal is feasible, and we may proceed to step 3.

Step 3: Compare z_1 found in step 1 to $c_2^T w^* + z_2^*$ where $z_2^* = (b - A_2 w^*)^T \lambda^*$ is found in step 2.

In step 1, solution of the master problem resulted in $z_1^*=28$

In step 2, solution of the subproblem resulted in $z_2^*=14$.

In both problems, $c_2=5$, and we found (master) or used (sub) $w^*=2$.

Benders optimal rule is

$$\underbrace{c_2^T w^*}_{\text{from master problem}} + \underbrace{(b - A_2 w^*)^T \lambda^*}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{z_1^*}_{\text{from master problem}}$$

Substitution yields:

$$\underbrace{5 \bullet 2}_{\text{from master problem}} + \underbrace{14}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{28}_{\text{from master problem}}$$

Benders optimal rule is not satisfied, we need to obtain the optimality cut from $z_2^* \leq (b - A_2 w^*)^T \lambda^*$. With

$$b = \begin{bmatrix} 12 \\ 12 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \quad \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 1.5 \end{bmatrix}$$

$$z_2^* \leq \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w \right)^T \begin{bmatrix} 0.5 \\ 1.5 \end{bmatrix} = [12 - w \quad 12 - 3w] \begin{bmatrix} 0.5 \\ 1.5 \end{bmatrix} = 24 - 5w$$

Now we return to step 1.

Step 1: Adjoin the optimality cut to Q , resulting in the following master problem:

$$\begin{aligned} \max \quad & z_1 = 5w + (z_2^*) \\ \text{subject to: } \quad & Q: w \leq 20, w \leq 4, z_2^* \leq M, z_2^* \leq 36 - 9w, z_2^* \leq 24 - 2w, z_2^* \leq 24 - 5w \\ & w \geq 0, w \text{ integer} \end{aligned}$$

This all-integer program can be solved using a branch and bound algorithm (both CPLEX and Matlab have one), but the solution can be identified using enumeration, since w can only be 0, 1, 2, 3, or 4.

Enumerating the solutions to this problem results in

$$\begin{aligned} w=0: z_2^* &= 24, z_1^* = 24 \\ w=1: z_2^* &= 19, z_1^* = 24 \\ w=2: z_2^* &= 14, z_1^* = 24 \\ w=3: z_2^* &= 9, z_1^* = 24 \\ w=4: z_2^* &= 0, z_1^* = 20 \end{aligned}$$

We see that $w=0, 1, 2,$ and 3 are equally good solutions

Steps 2 and 3: for each of these solutions, using the value of w found in the master, get the dual. Then check Benders rule. The general form of the dual is below.

$$\begin{aligned} \min z_2 &= (b - A_2 w^*)^T \lambda \\ \text{s.t.} \quad & A_1^T \lambda \geq c_1 \\ & \lambda \geq 0 \end{aligned} \quad \rightarrow \quad \begin{aligned} \min z_2 &= \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} w^* \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix} \\ & \lambda_1, \lambda_2 \geq 0 \end{aligned}$$

$$\text{Benders optimal rule is } \underbrace{c_2^T w^*}_{\text{from master problem}} + \underbrace{(b - A_2 w^*)^T \lambda^*}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{z_1^*}_{\text{from master problem}}$$

$w^*=0$, the subproblem becomes:

$$\min z_2 = \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} 0 \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 12\lambda_1 + 2\lambda_2$$

s.t.

$$\begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$\lambda_1, \lambda_2 \geq 0$$

Solution from CPLEX is $\lambda_1=2$, $\lambda_2=0$, with objective function value $z_2^*=24$.

Benders rule:
$$\underbrace{5 \bullet 0}_{\text{from master problem}} + \underbrace{24}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{24}_{\text{from master problem}}$$

This solution is optimal. Dual variables obtained from CPLEX are $x_1=6$, $x_2=0$.

$w^*=1$, the subproblem becomes:

$$\min z_2 = \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} 1 \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 11\lambda_1 + 9\lambda_2$$

s.t.

$$\begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$\lambda_1, \lambda_2 \geq 0$$

Solution from CPLEX is $\lambda_1=0.5$, $\lambda_2=1.5$, with objective function value $z_2^*=19$.

Benders rule:
$$\underbrace{5 \bullet 1}_{\text{from master problem}} + \underbrace{19}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{24}_{\text{from master problem}}$$

This solution is optimal. Dual variables obtained from CPLEX are $x_1=4$, $x_2=1$.

$w^*=2$, the subproblem becomes:

$$\min z_2 = \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} 2 \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 10\lambda_1 + 6\lambda_2$$

s.t.

$$\begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$\lambda_1, \lambda_2 \geq 0$$

Solution from CPLEX is $\lambda_1=0.5$, $\lambda_2=1.5$, with objective function value $z_2^*=14$.

Benders rule:
$$\underbrace{5 \bullet 2}_{\text{from master problem}} + \underbrace{14}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{24}_{\text{from master problem}}$$

This solution is optimal. Dual variables obtained from CPLEX are $x_1=2$, $x_2=2$.

$w^*=3$, the subproblem becomes:

$$\min z_2 = \left(\begin{bmatrix} 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 1 \\ 3 \end{bmatrix} 3 \right)^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 9\lambda_1 + 3\lambda_2$$

s.t.

$$\begin{bmatrix} 2 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \geq \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$\lambda_1, \lambda_2 \geq 0$$

Solution from CPLEX is $\lambda_1=0$, $\lambda_2=3$, with objective function value $z_2^*=9$.

Benders rule:
$$\underbrace{5 \bullet 3}_{\text{from master problem}} + \underbrace{9}_{z_2^* \text{ from subproblem}} \stackrel{?}{=} \underbrace{24}_{\text{from master problem}}$$

This solution is optimal. Dual variables obtained from CPLEX are $x_1=0$, $x_2=3$.

Problem summary:

Recall our original problem:

$$\begin{aligned} \max \quad & z_1 = 4x_1 + 3x_2 + 5w \\ \text{subject to:} \quad & 2x_1 + 3x_2 + w \leq 12 \\ & 2x_1 + x_2 + 3w \leq 12 \\ & w \leq 20 \\ & x_1, x_2, w \geq 0, \\ & w \text{ integer} \end{aligned}$$

Optimal solutions to this problem result in an objective function value of $z_1=24$ and are:

- $w=0, x_1=6, x_2=0$
- $w=1, x_1=4, x_2=1$
- $w=2, x_1=2, x_2=2$
- $w=3, x_1=0, x_2=3$

Some comments about this problem:

1. It is coincidence that the values of x_1 and x_2 for the optimal solution also turn out to be integers.
2. The fact that there are multiple solutions is typical of MIP problems. MIP problems are generally non-convex.

5.0 Benders simplifications

In the previous section, we studied problems having the following structure:

$$\begin{aligned} \max z_1 &= c_1^T x + c_2^T w \\ \text{s.t.} \quad & Dw \leq e \\ & A_1 x + A_2 w \leq b \\ & x, w \geq 0 \\ & w \text{ integer} \end{aligned}$$

and we defined the master problem and primal subproblem as

Master :

$$\max z_1 = c_2^T w + z_2^*$$

s.t.

$$Dw \leq e$$

$$w \geq 0$$

w integer

Primal subproblem :

$$\max z_2 = c_1^T x$$

s.t.

$$A_1 x \leq b - A_2 w^*$$

$$x \geq 0$$

However, what if our original problem appears as below, which is the same as the original problem except that it does not contain an “x” in the objective function, although the “x” still remains in one of the constraints.

$$\max z_1 = c_2^T w$$

s.t.

$$Dw \leq e$$

$$A_1 x + A_2 w \leq b$$

$$x, w \geq 0$$

w integer

In this case, the master problem and the primal subproblem become:

Master :

$$\max z_1 = c_2^T w$$

s.t.

$$Dw \leq e$$

$$w \geq 0$$

w integer

Primal subproblem :

$$\max z_2 = ???$$

s.t.

$$A_1 x \leq b - A_2 w^*$$

$$x \geq 0$$

One sees clearly here that the primal subproblem has no z_2 to maximize! One way to address this issue is to introduce a vector of non-negative slack variables s having one element for each constraint. We will minimize the sum of these slack variables, so that a non-zero value of this sum indicates the subproblem is

infeasible. That is, we replace our primal subproblem with a feasibility check subproblem, as follows:

Master :

$$\begin{aligned} \max z_1 &= c_2^T w \\ \text{s.t.} \\ Dw &\leq e \\ w &\geq 0 \\ w &\text{ integer} \end{aligned}$$

Feasibility check subproblem :

$$\begin{aligned} \min v &= \text{Ones}^T s \\ \text{s.t.} \\ A_1 x - s &\leq b - A_2 w^* \\ x &\geq 0, \quad s \geq 0 \end{aligned}$$

Here, *Ones* is a column vector of 1's, so that $v = \text{Ones}^T s$ is the summation of all elements in the column vector s . When $v=0$, each constraint in $A_1 x - s \leq b - A_2 w^*$ is satisfied so that $A_1 x \leq b - A_2 w^*$, which means the constraints to the original problem are in fact satisfied.

In this case, one observes that if $v=0$, then the problem is solved since Benders optimality rule will always be satisfied.

$$\underbrace{c_2^T w^*}_{\text{from master problem}} + \underbrace{(b - A_2 w^*)^T \lambda^*}_{z_2^* \text{ from subproblem}} = \underbrace{z_1^*}_{\text{from master problem}}$$

Here, z_1 is always zero, and the other two terms come from the master problem, therefore if the problem is feasible, it is optimal, and no step 3 is necessary.

One question does arise, however, and that is what should be the feasibility cuts returned to the master problem if the feasibility check subproblem results in $v > 0$? The answer to this is stated in [7] and shown in [8] to be

$$v + \lambda A_2 (w^* - w) < 0$$

This kind of problem is actually very common. Figure 4, using a SCOPF to motivate decomposition methods for enhancing computational efficiency, is of this type. This is very similar to the so-called simultaneous feasibility test (SFT) of industry.

The SFT (Simultaneous Feasibility Test) is widely used in SCED and SCUC [9,10, 11]. SFT is a contingency analysis process. The objective of SFT is to determine violations in all post-contingency states and to produce generic constraints to feed into economic dispatch or unit commitment, where a generic constraint is a transmission constraint formulated using linear sensitivity coefficients/factors.

The ED or UC is first solved without considering network constraints and security constraints. The results are sent to perform the security assessment in a typical power flow. If there is an existing violation, the new constraints are generated using the sensitivity coefficients/ factors and are added to the original problem to solve repetitively until no violation exists. The common flowchart is shown in Fig. 6.

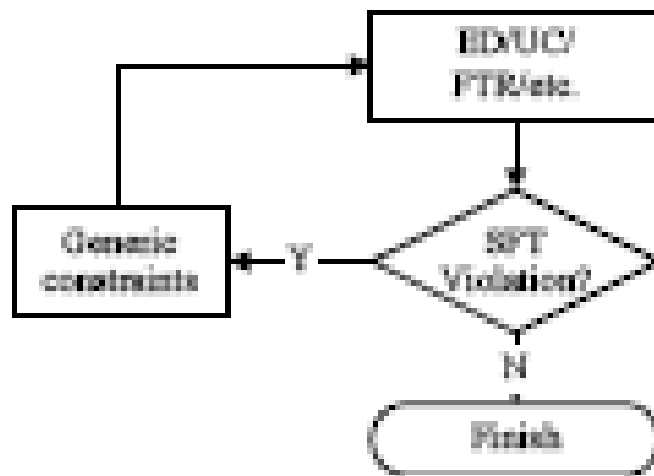


Fig. 6

This section has focused on the very common case where the general Benders approach degenerates to a feasibility test problem only, i.e., the optimality test does not need to be done. There are at least three other degenerate forms of Benders:

- **No feasibility problem**: In some situations, the optimality problem will be always feasible, and so the feasibility problem is unnecessary.

- **Dual-role feasibility and optimality problem**: In some applications, the feasibility and optimality problem can be the same problem.

Reference [7] provides examples of these degenerate forms of Benders decomposition.

6.0 Application of Benders to other Problem Types

This section is best communicated by quoting from Geoffrion [4] (highlight added), considered the originator of Generalized Benders.

“J.F. Benders devised a clever approach for exploiting the structure of mathematical programming problems with complicating variables (variables which, when temporarily fixed, render the remaining optimization problem considerably more tractable). For the class of problems specifically considered by Benders, fixing the values of the complicating variables reduces the given problem to an ordinary linear program, parameterized, of course, by the value of the complicating variables vector. The algorithm he proposed for finding the optimal value of this vector employs a cutting-plane approach for building up adequate representations of (i) the extremal value of the linear program as a function of the parameterizing vector and (ii) the set of values of the parameterizing vector for which the linear program is feasible. Linear programming duality theory was employed to derive the natural families of cuts characterizing these representations, and the parameterized linear program itself is used to generate what are usually deepest cuts for building up the representations.

In this paper, Benders' approach is generalized to a broader class of programs in which the parametrized subproblem need no longer be a linear program. Nonlinear convex duality theory is employed to derive the natural families of cuts corresponding to those in Benders' case. The conditions under which such a generalization is possible and appropriate are examined in detail.”

The spirit of the above quotation is captured by the below modified formulation of our problem.

The problem can be generally specified as follows:

$$\begin{aligned} \max z_1 &= f(x) + c_2^T w \\ \text{s.t.} & \\ Dw &\leq e \\ F(x) + A_2 w &\leq b \\ x, w &\geq 0 \\ w &\text{ integer} \end{aligned}$$

Define the master problem and primal subproblem as

Master :

$$\begin{aligned} \max z_1 &= c_2^T w + z_2^* \\ \text{s.t.} & \\ Dw &\leq e \\ w &\geq 0 \\ w &\text{ integer} \end{aligned}$$

Primal subproblem :

$$\begin{aligned} \max z_2 &= f(x) \\ \text{s.t.} & \\ F(x) &\leq b - A_2 w^* \\ x &\geq 0 \end{aligned}$$

The Benders process must be generalized to solve the above problem since the subproblem is a nonlinear program (NLP) rather than a linear program (LP). Geoffrion shows how to do this [4].

In the above problem, w is integer, the master is therefore a linear integer program (LIP); the complete problem is therefore an integer NLP. If Benders can solve this problem, then it will also solve the problem when w is continuous, so that the master is LP and subproblem is NLP. If this is the case, then Benders will also solve the problem where both master and subproblem are LP, which is a very common approach to solving very-large-scale linear programs. Table 1 summarizes the various problems Benders is known to be able to solve.

Table 1

		Master	
		ILP	LP
Subproblem	LP	√	√
	NLP	√	√

One might ask whether Benders can handle a nonlinear integer program in the master, but it is generally unnecessary to do so since such problems can usually be decomposed to an ILP master with a NLP subproblem.

7.0 Application of Benders to Stochastic Programming

For good, but brief overviews of Stochastic Programming, see [12] and [13].

In our example problem, we considered only a single subproblem, as shown below.

$$\begin{aligned}
 \max z_1 &= c_1^T x + c_2^T w \\
 s.t. & \\
 Dw &\leq e \\
 A_1 x + A_2 w &\leq b \\
 x, w &\geq 0 \\
 w &\text{ integer}
 \end{aligned}$$

To prepare for our generalization, we rewrite the above in a slightly different form, using slightly different notation:

$$\begin{aligned}
 \max z_1 &= c^T w + d_1^T x_1 \\
 s.t. & \\
 Dw &\leq e \\
 B_1 w + A_1 x_1 &\leq b_1 \\
 x_1, w &\geq 0 \\
 w &\text{ integer}
 \end{aligned}$$

Now we are in position to extend our problem statement so that it includes more than a single subproblem, as indicated in the structure provided below.

$$\max z_1 = c^T w + d_1^T x_1 + d_2^T x_2 + \dots + d_n^T x_n$$

s.t.

$$Dw \leq e$$

$$B_1 w + A_1 x_1 \leq b_1$$

$$B_2 w + A_2 x_2 \leq b_2$$

$$\vdots \leq \vdots$$

$$B_m w + A_n x_n \leq b_n$$

$$x_k, w \geq 0$$

In this case, the master problem is

$$\max z_1 = c^T w + \sum_{i=1}^n z_i(x_i)$$

s.t.

$$Dw \leq e$$

$$w \geq 0$$

where z_i provide values of the maximization subproblem given by:

$$\max z_i = d_i^T x_i$$

s.t.

$$A_i x_i \leq b_i - B_i w$$

$$x_i \geq 0$$

Note that the constraint matrix for the complete problem appears as:

$$\begin{bmatrix} D \\ B_1 & A_1 \\ B_2 & & A_2 & \ddots \\ \vdots & & & & \\ B_n & & & & A_n \end{bmatrix} \begin{bmatrix} w \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \leq \begin{bmatrix} e \\ b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

The constraint matrix shown above, if one only considers D , B_1 , and A_1 , has an L-shape, as indicated below.

$$\begin{array}{c}
 \boxed{\begin{array}{cc} D & \\ B_1 & A_1 \end{array}} \\
 B_2 \quad A_2 \\
 \vdots \quad \ddots \\
 B_n \quad A_n
 \end{array}
 \begin{array}{c}
 \left[\begin{array}{c} w \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \right] \leq \left[\begin{array}{c} e \\ b_1 \\ b_2 \\ \vdots \\ b_n \end{array} \right]
 \end{array}$$

Consequently, methods to solve these kinds of problems, when they are formulated as stochastic programs, are called L-shaped methods.

But what is, exactly, a stochastic program [12]?

- A stochastic program is an optimization approach to solving decision problems under uncertainty where we make some choices for “now” (the current period) represented by w , in order to minimize our present costs.
- After making these choices, event i happens, so that we take *recourse*², represented by x , in order to minimize our costs under each event i that could occur in the next period.
- Our decision must be made *a-priori*, however, and so we do not know which event will take place, but we do know that each event i will have probability p_i .
- Our goal, then, is to minimize the cost of the decision for “now” (the current period) plus the expected cost of the later recourse decisions (made in the next period).

² Recourse is the act of turning or applying to a person or thing for aid.

A good application of this problem for power systems is the security-constrained optimal power flow (SCOPF) with corrective action.

- In this problem, we dispatch generation to minimize costs for the network topology that exists in this 15 minute period. Each unit generation level is a choice, and the complete decision is captured by the vector w . The dispatch costs are represented by $c^T w$.
- These “normal” conditions are constrained by the power flow equations and by the branch flow and unit constraints, all of which are captured by $Dw \leq e$.
- Any one of $i=1, \dots, n$ contingencies may occur in the next 15 minute period. Given that we are operating at w during this period, each contingency i requires that we take corrective action (modify the dispatch, drop load, or reconfigure the network) specified by x_i .
- The cost of the corrective action for contingency i is $d_i^T x_i$, so that the expected costs over all possible contingencies is $\sum p_i d_i^T x_i$.
- Each contingency scenario is constrained by the post-contingency power flow equations, and by the branch flow and unit constraints, represented by $B_i w + A_i x_i \leq b_i$. The dependency on w (the pre-contingency dispatch) occurs as a result of, for example, unit ramp rate limitations.

A 2-stage recourse problem is formulated below:

$$\begin{aligned} \min z_1 &= c^T w + \sum_{i=1}^n p_i d_i^T x_i \\ \text{s.t.} & \\ Dw &\leq e \\ B_1 w + A_1 x_1 &\leq b_1 \\ B_2 w + A_2 x_2 &\leq b_2 \\ &\vdots \\ B_m w + A_n x_n &\leq b_n \\ x, w &\geq 0 \end{aligned}$$

where p_i is the (scalar) probability of event i , and d_i is the vector of costs associated with taking recourse action x_i . Each constraint equation $B_i w + A_i x_i \leq b_i$ limits the recourse actions that can be taken in response to event i , and depends on the decisions w made for the current period.

Formulation of this problem for solution by Benders (the L-shaped method) results in the master problem as

$$\begin{aligned} \min z_1 &= c^T w + \sum_{i=1}^n z_i \\ \text{s.t.} & \\ D w &\leq e \\ w &\geq 0 \end{aligned}$$

where z_i is minimized in the subproblem given by:

$$\begin{aligned} \min z_i &= p_i d_i^T x_i \\ \text{s.t.} & \\ A_i x_i &\leq b_i - B_i w \\ x_i &\geq 0 \end{aligned}$$

Note that the first-period decision, w , does not depend on which second-period scenario actually occurs (but does depend on a probabilistic weighting of the various possible futures). This is called the *nonanticipativity property*. The future is uncertain and so today's decision cannot take advantage of knowledge of the future.

Recourse models can be extended to handle multistage problems, where a decision is made “now” (in the current period), we wait for some uncertainty to be resolved, and then we make another decision based on what happened. The objective is to minimize the expected costs of all decisions taken. This problem can be appropriately thought of as the coverage of a decision tree, as shown in Fig. 7, where each “level” of the tree corresponds to another stochastic program.

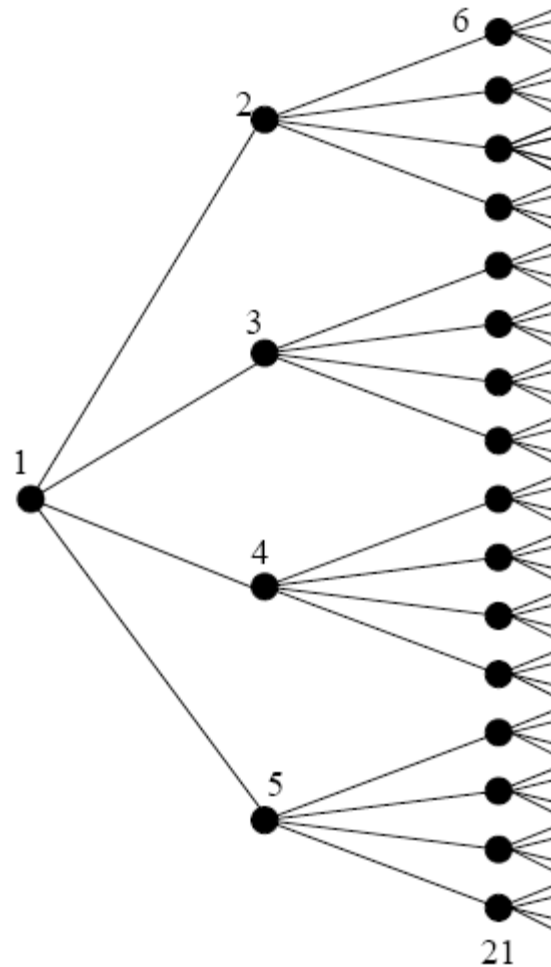


Fig. 7

Multistage stochastic programs have been applied to handle uncertainty in planning problems before. This is a reasonable approach; however, one should be aware that computational requirements increase with number of time periods and number of scenarios (contingencies) per time period. Reference [14] by J. Beasley provides a good, but brief overview of multistage stochastic programming. Reference [15], notes for an entire course, provides a comprehensive treatment of stochastic programming including material on multistage stochastic programming.

8.0 Two related problem structures

We found the general form of the stochastic program to be

$$\max z_1 = c^T w + d_1^T x_1 + d_2^T x_2 + \dots + d_n^T x_n$$

s.t.

$$Dw \leq e$$

$$B_1 w + A_1 x_1 \leq b_1$$

$$B_2 w + A_2 x_2 \leq b_2$$

$$\vdots \leq \vdots$$

$$B_m w + A_n x_n \leq b_n$$

$$x_k, w \geq 0$$

so that the constraint matrix appears as

$$\begin{bmatrix} D & & & & \\ B_1 & A_1 & & & \\ B_2 & & A_2 & & \\ \vdots & & & \ddots & \\ B_n & & & & A_n \end{bmatrix} \begin{bmatrix} w \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \leq \begin{bmatrix} e \\ b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

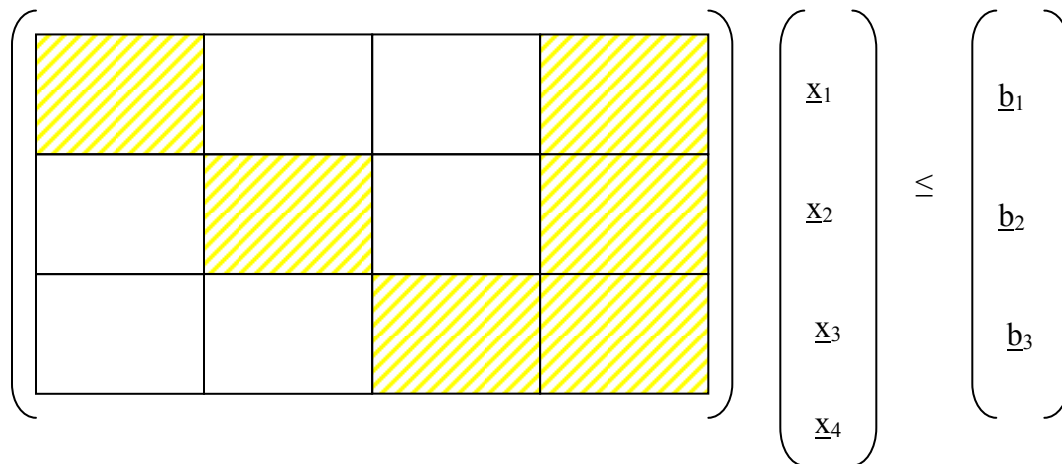
If we move the w vector to the bottom of the decision vector, the constraint matrix appears as

$$\begin{bmatrix} A_1 & & & B_1 \\ & A_2 & & B_2 \\ & & \ddots & \vdots \\ & & & A_n & B_n \\ & & & & D \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ w \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \\ e \end{bmatrix}$$

Notice that this is the structure that we introduced in Fig. 3 at the beginning of these notes, repeated here for convenience. We referred to this structure as “*block angular with linking variables.*” Now we will call it the *Benders structure*.

This means that coupling exists between what would otherwise be independent optimization problems via various parts of the problem occur via variables, in this case w .

For example, in the SCOPF with corrective action, the almost-independent optimization problems are the n optimization problems related to the n contingencies. The dependency occurs via the dispatch determined for the existing (no-contingency) condition.



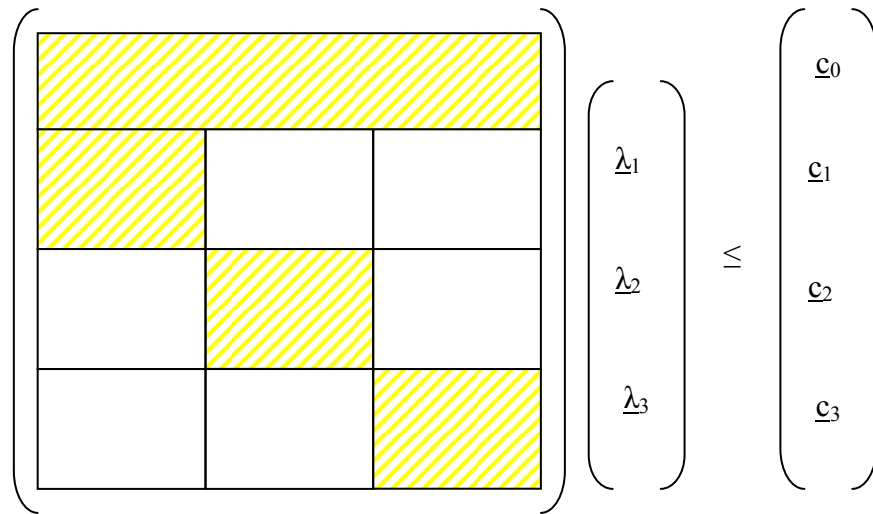
A simpler example harkens back to our illustrations of the CEO and his or her organization. In this case, the CEO must decide on how to allocate a certain number of employees to each department, then each department minimizes costs (or maximizes profits) subject to constraints on financial resources.

Recall from our discussion of duality that one way that primal and dual problems are related is that

- coefficients of one variable across multiple primal constraints
 - are coefficients of multiple variables in one dual constraint,
- as illustrated below.

<p><u>Problem P</u></p> $\max F = 3x_1 + 5x_2$ <p>s.t.</p> $\begin{aligned} x_1 &\leq 4 \\ 2x_2 &\leq 12 \\ 3x_1 + 2x_2 &\leq 18 \\ x_1 \geq 0, x_2 &\geq 0 \end{aligned}$ <p style="text-align: center;">Primal Problem</p>	<p><u>Problem D</u></p> $\min G = 4\lambda_1 + 12\lambda_2 + 18\lambda_3$ <p>subject to</p> $\begin{aligned} \lambda_1 + 3\lambda_3 &\geq 3 \\ 2\lambda_2 + 2\lambda_3 &\geq 5 \\ \lambda_1 \geq 0, \lambda_2 \geq 0, \lambda_3 &\geq 0 \end{aligned}$ <p style="text-align: center;">Dual Problem</p>
--	--

This can be more succinctly stated by saying that the dual constraint matrix is the transpose of the primal constraint matrix. You should be able to see, then, that the structure of the dual problem to a problem with the Benders structure looks like Fig. 2, repeated here for convenience.



This structure differs from that of the Benders structure (where the subproblems were linked by variables) in that now, the subproblems are linked by constraints. This problem is actually solved most effectively by another algorithm called Dantzig-Wolfe (DW) decomposition. We will refer to the above structure as the DW structure.

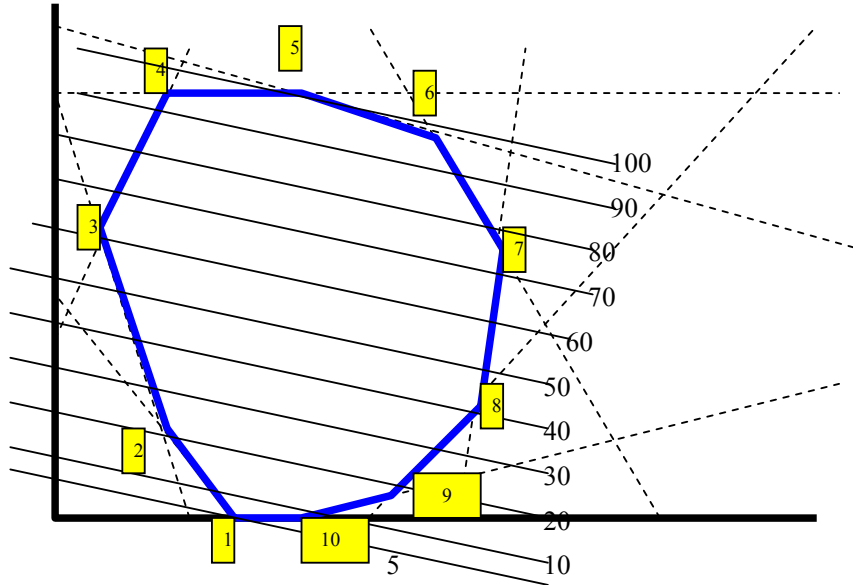


Fig. 8

Observation 1: Each individual subproblem represented by

$$\begin{aligned} \max z &= c_i^T x_i \\ \text{subject to } &A_{ii}x_i = b_i \end{aligned}$$

has its own set of extreme points. We refer to these extreme points as the *subproblem i extreme points*, denoted by x_i^k , $k=1, \dots, p_i$ where p_i is the number of extreme points for subproblem i .

Observation 2: Corresponding to each extreme point solution x_i^k , the amount of corporate resources used is $A_{0i}x_i^k$, an $m_0 \times 1$ vector.

Observation 3: The contribution to the objective function of the extreme point solution is $c_i^T x_i^k$, a scalar.

Definition 2: A convex combination of extreme points is a point

$$\sum_{k=1}^{p_i} x_i^k y_i^k, \text{ where } \sum_{k=1}^{p_i} y_i^k = 1, \text{ so that } y_i^k \text{ is the fraction of extreme point}$$

x_i^k in the convex combination. Figure 9 below shows a white dot in the interior of the region illustrating a convex combination of the two extreme points numbered 4 and 9.

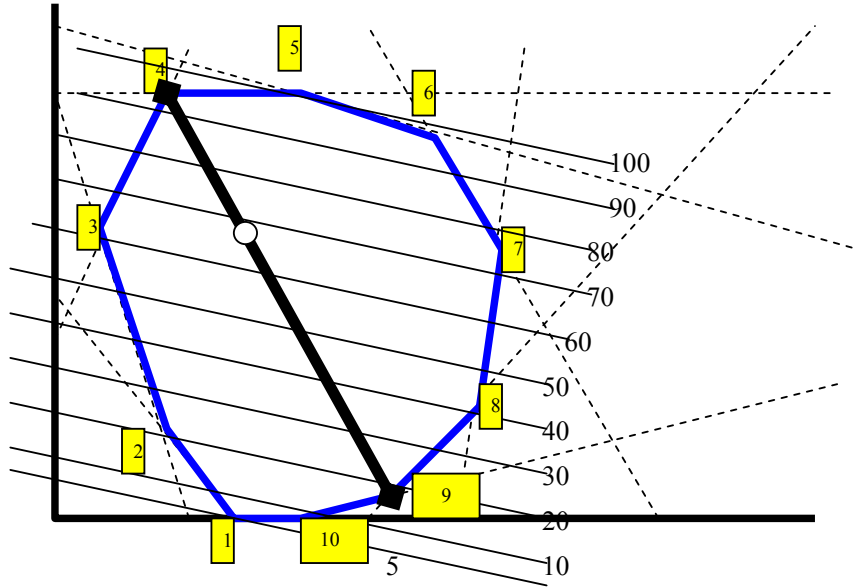


Fig. 9

Fact 1: Any convex combination of extreme points must be feasible to the problem. This should be self-evident from Fig. 9 and can be understood from the fact that the convex combination is a “weighted average” of the extreme points and therefore must lie “between” them (for two extreme points) or “interior to” them (for multiple extreme points).

Fact 2: Any point in the feasible region may be identified by appropriately choosing the y_i^k .

Observation 4a: Since the convex combination of extreme points is a weighted average of those extreme points, then the total resource usage by that convex combination will also be a weighted average

of the resource usage of the extreme points, i.e., $\sum_{k=1}^{p_i} (A_{0i} x_i^k) y_i^k$. The

total resources for the complete problem is the summation over all

of the subproblems, $\sum_{i=1}^h \sum_{k=1}^{p_i} (A_{0i} x_i^k) y_i^k$.

Observation 4b: Since the convex combination of extreme points is a weighted average of those extreme points, then the contribution to the objective function contribution by that convex combination will also be a weighted average of the objective function contribution of the extreme points, i.e., $\sum_{k=1}^{p_i} (c_i^T x_i^k) y_i^k$. The total objective function can

be expressed as the summation over all subproblems, $\sum_{i=1}^h \sum_{k=1}^{p_i} (c_i^T x_i^k) y_i^k$.

Based on Observations 4a and 4b, we may now transform our optimization problem P as a search over all possible combinations of points within the feasible regions of the subproblems to maximize the total objective function, subject to the constraint that the y_i^k must sum to 1.0 and must be nonnegative, i.e.,

$$\begin{aligned}
 \text{P-T} \quad & \max z = \sum_{i=1}^h \sum_{k=1}^{p_i} (c_i^T x_i^k) y_i^k \\
 & \text{subject to } A_{00} x_0 + \sum_{i=1}^h \sum_{k=1}^{p_i} (A_{0i} x_i^k) y_i^k = b_0 \quad m_0 \text{ constraints} \\
 & \sum_{k=1}^{p_i} y_i^k = 1, \quad i = 1, \dots, h \quad h \text{ constraints} \\
 & y_i^k \geq 0, \quad i = 1, \dots, h, \quad k = 1, \dots, p_i
 \end{aligned}$$

Note that this new problem P-T (P-transformed) has m_0+h constraints, in contrast to problem P which has $\sum_{i=0}^h m_i$ constraints.

Therefore it has far fewer constraints. However, whereas problem P has only $\sum_{i=0}^h n_i$ variables, this new problem has as many variables as it

has total number of extreme points across the h subproblems, $\sum_{i=0}^h p_i$, and so it has a much larger number of variables.

The DW decomposition method solves the new problem without explicitly considering all of the variables.

Understanding the DW method requires having a background in linear programming so that one is familiar with the revised simplex algorithm. We do not have time to cover this algorithm in this class, but it is standard in any linear programming class to do so.

Instead, we provide an economic interpretation to the DW method.

In the first constraint of problem P-T, b_0 can be thought of as representing shared resources among the various subproblems $i=1, \dots, h$.

Let the first m_0 dual variables of problem P-T be contained in the $m_0 \times 1$ vector π . Each of these dual variables provide the change in the objective as the corresponding right-hand-side (a resource) is changed.

→ That is, if b_{0k} is changed by $b_{0k} + \Delta$, then the optimal value of the objective is modified by adding $\pi_k \Delta$.

→ Likewise, if the i^{th} subproblem (department) increases its use of resource b_{0k} by Δ (instead of increasing the amount of the resource by Δ), then we can consider that that subproblem (department) has incurred a “charge” of $\pi_k \Delta$. This “charge” worsens its contribution to the complete problem P-T objective, and accounting for all shared resources b_{0k} , $k=1, m_0$, the contribution to the objective is $c_i^T x_i - \pi^T A_{0i} x_i$ where $A_{0i} x_i$ is the amount of shared resources consumed by the i^{th} subproblem (department).

One may think of these dual variables contained in π as the “prices” that each subproblem (department) must pay for use of the corresponding shared resources.

Assuming each subproblem $i=1, \dots, h$ represents a different department in the CEO's organization, the DW-method may be interpreted in the following way, paraphrased from [2]:

- If each department i worked independently of the others, then each would simply minimize its part of the objective function, i.e., $c_i^T x_i$.
- However, the departments are not independent but are linked by the constraints of using resources shared on a global level.
- The right-hand sides b_{0k} , $k=1, \dots, m_0$, are the total amounts of resources to be distributed among the various departments.
- The DW method consists of having the CEO make each department pay a unit price, π_k , for use of each resource k .
- Thus, the departments react by including the prices of the resources in its own objective function. In other words,
 - o Each department will look for new activity levels x_i which minimize $c_i^T x_i - \pi^T A_{0i} x_i$.
 - o Each department performs this search by solving the following problem:

$$\begin{aligned} \max \quad & c_i^T x_i - \pi^T A_{0i} x_i \\ \text{subject to} \quad & A_{ii} x_i = b_i \\ & x_{ik} \geq 0 \quad \forall k \end{aligned}$$

- The departments make proposals of activity levels x_i back to the CEO, and the CEO then determines the optimal weights y_i^k for the proposals by solving problem P-T, getting a new set of prices π , and the process repeats until all proposals remain the same.

Reference [2], p. 346, and [16], pp. 349-350, provide good articulations of the above DW economic interpretation.

10.0 Motivating interest in multicommodity flow problems

Multicommodity flow problems are optimization problems where one is interested in identifying the least cost method of moving two or more types of quantities (commodities) from certain locations where they are produced to certain locations where they are consumed, and the transport mechanism has the form of a network.

Clearly, multicommodity flow problems are of great interest in commodity transportation problems. For example, both the rail and trucking industries extensively utilize multicommodity. Airlines also use it (first-class and economy passengers may be viewed as two different “commodities”).

Although the transport mechanism for electric energy certainly has the form of a network (the transmission grid!), electric power planners have not traditionally been interested in multicommodity flow problems because of two issues:

- (a) Commodity flow (single or multiple) problems do not respect Kirchoff’s Voltage Law (they do respect nodal balance).
- (b) Transmission networks flow only 1 commodity, electric energy.

Issue (a) is one electric planners can live with in some cases when the flow on a branch is directly controllable. This is the case for DC lines and tends to be the case for AC intertie lines between different control areas. So-called network flow methods have therefore been deployed frequently to solve certain kinds of problems in electric planning. For example,

- Minimal cost network flow has been employed to solve problems combining fuel transportation (rail, natural gas) with electric energy transportation (transmission). Example applications include [17, 18].
- A maximum flow algorithm has been employed to solve the multiarea reliability problem for electric power systems. An extensive set of notes on this method is located at www.ee.iastate.edu/~jdm/ee653/U21-inclass.doc, Section U21.7.

Issue (b), however, has precluded most electric power system interest in multicommodity flows. That may need to change in the future. Why?

The reason is simple. Transportation systems, e.g., rail, truck (or highway), and airline systems have been almost totally decoupled from the electric system. There is fairly clear indication that this decoupled state is coming to an end in that the pluggable hybrid electric vehicle (PHEV) or the pluggable electric vehicle (PEV) is already becoming quite common, and projections are that electric utility grids are likely to see millions of such vehicles plugging into the grid within the next 5-10 years.

Although rail in the US is almost entirely powered by on-board diesel generators, both Europe and Japan have utilized electric trains extensively, indicating the technology is available. These electric trains can have speeds significantly higher than existing US rail, which typically travels no faster than 100 mph. Increasing that speed (doubling?) could attract increased passenger travel, resulting in decreased airline use.

These observations motivate an interest in consideration of the increased coupling, and thus interdependencies, between the electric system (including the fuel production and delivery systems) and the transportation system. Figure 10 illustrates these various systems.

These ideas motivate interest in the development of modeling methods that provide optimized investment plans for electric infrastructure and transportation infrastructure simultaneously. Such modeling is likely to make use of multicommodity transportation flow problems.

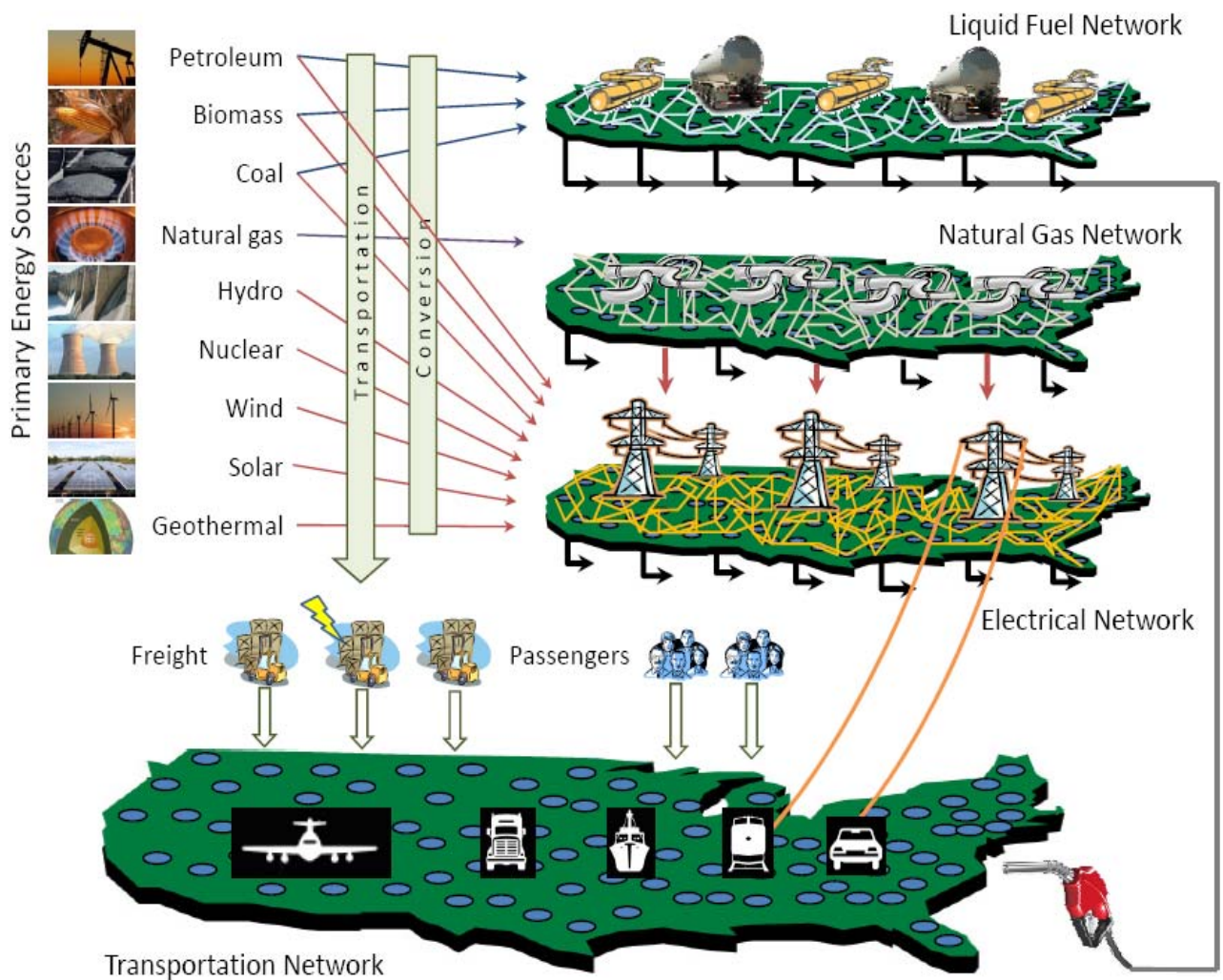


Fig. 10

11.0 Multicommodity flow problem: formulation

Multicommodity flow problems have the quality that the branches (arcs) between nodes may be capacitated, for each commodity flow, and for the sum total of all commodity flows. The below formulation is adapted from [16].

Suppose that we have a network with m nodes and n arcs in which there will flow t different commodities. We use the following nomenclature:

- u_i : vector of upper limits on flow for commodity i in the arcs of the network, so that u_{ipq} is the upper limit on the flow of commodity i in arc (p,q) .
- u : vector of upper limits on the sum of all commodities flowing in the arcs of the network, so that u_{pq} is the upper limit on the sum of all commodity flows in arc (p,q) .
- c_i : vector of arc costs in the network for commodity i , so that c_{ipq} is the unit cost of commodity i on arc (p,q) .
- b_i : vector of supplies (or demands) of commodity i in the network, so that b_{iq} is the supply (if $b_{iq} > 0$) or demand (if $b_{iq} < 0$) of commodity i at node q .
- A : node-arc incidence matrix of the network.
- x_i : vector of flows of commodity i in the network.

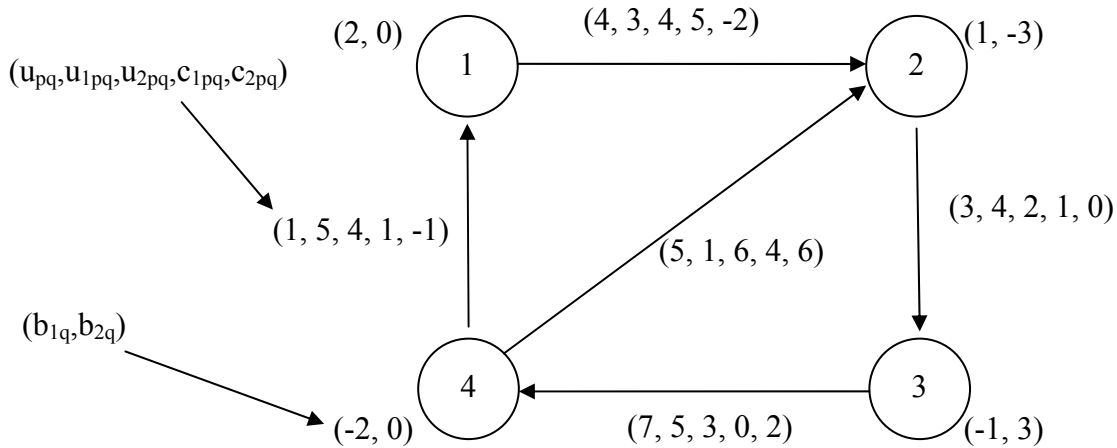
The linear programming formulation for the multicommodity minimal cost flow problem is as follows:

$$\begin{aligned} \min \quad & \sum_{i=1}^t c_i^T x_i \\ \text{subject to} \quad & \sum_{i=1}^t x_i \leq u \\ & Ax_i = b_i \quad i = 1, \dots, t \\ & 0 \leq x_i \leq u_i \quad i = 1, \dots, t \end{aligned}$$

Observe that the first constraint is across multiple commodities. The second constraint, actually a set of constraints, is each across a single commodity.

12.0 Multicommodity flow problem: example

This example is adapted from [16]. It is a 2-commodity minimal cost flow problem and is illustrated in Fig. 11.



The constraint matrix for this system is displayed below.

1	0	0	0	0	1	0	0	0	0	\leq	4	=	u_{12}
0	1	0	0	0	0	1	0	0	0		3		u_{23}
0	0	1	0	0	0	0	1	0	0		7		u_{34}
0	0	0	1	0	0	0	0	1	0		1		u_{41}
0	0	0	0	1	0	0	0	0	1		5		u_{42}
1	0	0	-1	0	0	0	0	0	0		2		b_{11}
-1	1	0	0	-1	0	0	0	0	0		1		b_{12}
0	-1	1	0	0	0	0	0	0	0		-1		b_{13}
0	0	-1	1	1	0	0	0	0	0		-2		b_{14}
0	0	0	0	0	1	0	0	-1	0		0		b_{21}
0	0	0	0	0	-1	1	0	0	-1	-3	b_{22}		
0	0	0	0	0	0	-1	1	0	0	3	b_{23}		
0	0	0	0	0	0	0	-1	1	1	0	b_{24}		

The first five equations represent constraints on total commodity flow and are less than or equal to their respective right-hand-sides. The last 8 equations represent nodal balance constraints for the four nodes, commodity 1, and the four nodes, commodity 2.

Observe that the structure is block-angular with linking constraints, i.e., this is what we called the GW-structure. We will therefore be able to effectively apply Dantzig-Wolfe decomposition to efficiently solve this problem.

-
- [1] F. Hillier and G. Lieberman, "Introduction to Operations Research," 4th edition, Holden-Day, Oakland California, 1986.
- [2] M. Minoux, "Mathematical Programming: Theory and Algorithms," Wiley, 1986.
- [3] J. Benders, "Partitioning procedures for solving mixed variables programming problems," *Numerische Mathematics*, 4, 238-252, 1962.
- [4] A. M. Geoffrion, "Generalized benders decomposition", *Journal of Optimization Theory and Applications*, vol. 10, no. 4, pp. 237–260, Oct. 1972.
- [5] S. Zions, "Linear and Integer Programming," Prentice-Hall, 1974.
- [6] J. Bloom, "Solving an Electricity Generating Capacity Expansion Planning Problem by Generalized Benders' Decomposition," *Operations Research*, Vol. 31, No. 1, January-February 1983.
- [7] Y. Li, "Decision making under uncertainty in power system using Benders decomposition," PhD Dissertation, Iowa State University, December 2008.
- [8] S. Granville, M. V. F. Pereira, G. B. Dantzig, B. Avi-Itzhak, M. Avriel, A. Monticelli, and L. M. V. G. Pinto, "Mathematical decomposition techniques for power system", Tech. Rep. 2473-6, EPRI, 1988.
- [9] D. Streiffert, R. Philbrick, and A. Ott, "A mixed integer programming solution for market clearing and reliability analysis", *Power Engineering Society General Meeting*, 2005. IEEE, pp. 2724–2731 Vol. 3, June 2005.
- [10] PJM Interconnection, "On line training materials", www.pjm.com/services/training/training.html.
- [11] ISO New England, "On line training materials", www.isone.com/support/training/courses/index.html.
- [12] www-fp.mcs.anl.gov/otc/Guide/OptWeb/continuous/constrained/stochastic/.
- [13] <http://users.iems.northwestern.edu/~jrbirge/html/dholmes/StoProIntro.html>.
- [14] <http://people.brunel.ac.uk/~mastjjb/jeb/or/sp.html>
- [15] <http://www.lehigh.edu/~jtl3/teaching/ie495/>
- [16] M. Bazaraa, J. Jarvis, and H. Sherali, "Linear Programming and Network Flows," second edition, Wiley, 1990.
- [17] A. Quelhas, E. Gil, J. McCalley, and S. Ryan, "A Multiperiod Generalized Network Flow Model of the U.S. Integrated Energy System: Part I – Model Description," *IEEE Transactions on Power Systems*, Volume 22, Issue 2, May 2007, pp 829 – 836.
- [18] A. Quelhas and J. McCalley, "A Multiperiod Generalized Network Flow Model of the U.S. Integrated Energy System: Part II – Simulation Results," *IEEE Transactions on Power Systems* Volume 22, Issue 2, May 2007, pp. 837 – 844.