A Linear Algorithm to Calculate Material Requirements for Processes Defined on a Table of Mixes and Ingredients

Alexander L. Kozachkov
Quantitative Developer
Bank of America Merrill Lynch
akozachkov@gmail.com

Leo A. Kozachkov
PhD Student
Massachusetts Institute of Technology
leokoz8@mit.edu

September 4, 2018

Abstract

Material requirements planning (MRP) is a production planning, scheduling, and inventory control system used to manage manufacturing processes. One of the major components of is the so-called bill of materials (BOM),which specifies the relationship between the components and the end product. MRP takes as input the information contained in the BOM [1]. In process industries (such as the food, beverage, chemical, pharmaceutical industries, etc.), the BOM is also known as the formula, or recipe. Process manufacturers, while mixing ingredients to make their end product, rely on a recipe or a formula. When a technological process does not involve chemical reactions and the number of steps is known, the formula, or recipe can be defined on a table containing the raw materials and mixes, along with the percentages of each raw material and each previously created mix used to create the current mix.

1 Problem Statement

Let's place the ingredients, including the mixes, in the table's rows and place the sequence of mixes in the table's columns, Figure 1.

	Mix 1	**Mix 2**	**Mix 3**	**Mix 4**
Raw Materials 1	10		25	
Raw Materials 2	20			20
Raw Materials 3		50		25
Raw Materials 4			25	5
Raw Materials 5	70	30		
Mix 1	**100**	20	25	20
Mix 2	x	**100**	25	10
Mix 3	x	x	**100**	20
Mix 4	x	x	x	**100**

Table 1: The very last mix is the target product.

The above table does not answer the question of how much of each ingredient must be used on each step to produce specific amount of target product. Neither it gives the exact amount of each raw material needed to start the technological process.

When the current mixes are not used as ingredients in the subsequent mixes, the algorithm for computing the required raw ingredients reduces to division, and is in fact trivial. However, it's much more complicated to build the general case for processes in which mixes can be used as ingredients in subsequent mixes. The problem can be formulated as a Linear Program [2] and thus can be solved by the efficient-in-practice Simplex Algorithm [3]. It was shown that the worst-case complexity of the Simplex Algorithm is exponential time[4]. Polynomial algorithms were invented in the late 1970s [5] and 1980s, including the first reasonably efficient algorithm by Karmarkar[6].

For this particular problem, there is a linear algorithm that builds another table using the original table and amount of target product as its input. The algorithm is implicitly based on material balance: matter is neither created nor destroyed in industrial non-radioactive processes.

2 Main Results

2.1 Definition

T is the amount of the product that we are building.

Let's assume that technological process consists of m steps (mixes) and requires n raw materials. They form $(n+m) \times m$ table, $\mathbf{P}$. The $(i,j)^{th}$ element of this table, $\mathbf{P}_{ij}$, is the percentage of an ingredient i used in mix j. For any column j the following condition is true (see Figure 2):

$$\sum_{i=1}^{n+j-1} P_{ij} = 100 \tag{1}$$

	1	**...**	**j**	**...**	**m**
1			P_{1j}		
...			...		
i			P_{ij}		
...			...		
n + j -1			P_{n+j-1j}		
n + j			100		
...			x		
n + m			x		

This represent the fact that total percentages of the ingredients including previously built mixes in the current mix is equal to 100 percent($\mathbf{P}_{n+j,j} = 100$).

Remaining elements of the column ($i = n+j+1$ to $m+n$) are not defined since any mix can be used on subsequent mixes only.

A different table being built is Target Table; each $\mathbf{A}_{ij}$ element of the table is an amount of an ingredient i used in the mix j.

It is not a good idea to try calculating each $\mathbf{A}_{ij}$ individually since any such formula would exponentially depend on the number $(n-j)$ of remaining steps in the technological process.

However, there is a simple fact that lets us calculate the table at once starting from the last mix and moving backwards.

2.2 Statement 1

$$\forall j > n, \mathbf{A}_{n+j,j} = \sum_{k=j+1}^{m} \mathbf{A}_{n+j,j} \tag{2}$$

	1	**...**	**j**	**...**	**m**
1					
⋮					
n+j			$A_{n+j,j}$		$A_{n+j,m}$
⋮			x		
n+m			x		

To prove this point, it's enough to say that amount of each mix created $\mathbf{A}_{n+j,j}$ must be equal to the total amounts of this mix used on subsequent steps (material balance).

In the example from Figure 1, assuming the target amount is 1000, the Target Table is shown on Figure 4. Note that each amount in bold is equal to additions of all others numbers from the raw, such as 280=30+50+200

Basically, calculation of current mix is done by finding target amount of the mix (using Statement 1) and proportions of the ingredients.

	Mix 1	Mix 2	Mix 3	Mix 4
Raw Materials 1	28		50	
Raw Materials 2	56			200
Raw Materials 3		75		250
Raw Materials 4			50	50
Raw Materials 5	196	45		
Mix 1	**280**	30	50	200
Mix 2	x	**150**	50	100
Mix 3	x	x	**200**	200
Mix 4	x	x	x	**1000**

2.3 Algorithm

Step 1: Calculate the last mix. Since last mix is the target product, amount of the last mix to build equals to the target amount. It means that

$$\forall i = 1 \text{ to } n+m-1, \mathbf{A}_{i,m} = T_1 * \frac{\mathbf{P}_{i,m}}{100} \tag{3}$$

Applying this formula to the 4^{th} column: (1,4) = (0/100) *1000 = 0; (2,4) = (20/100) *1000 = 200; (3,4) = (25/100) *1000 = 250; (4,4) = (5/100) *1000 = 50; (5,4) = (0/100) *1000 = 0; (6,4) = (20/100) *1000 = 200; (7,4) = (10/100) *1000 = 100; (8,4) = (20/100) *1000 = 200;

Step 2: Calculate second to last mix. The entire amount of this mix will be used in the last mix so it equals to the previously calculated $\mathbf{A}_{n+m-1,m} = T_2$ Since it is previously calculated - we can use a formula similar to the (3).

$$\forall i = 1 \text{ to } n+m-2, \mathbf{A}_{i,m-1} = T_2 \frac{\mathbf{P}_{i,m-1}}{100} \tag{4}$$

Applying this formula to the 3^{rd} column: (1,3) = (25/100) *200 = 50; (2,3) = (0/100) *200 = 0; (3,3) = (0/100) *200 = 0; (4,3) = (25/100) *200 = 50; (5,3) = (0/100) *200 = 0; (6,3) = (25/100) *200 = 50; (7,3) = (25/100) *200 = 50;

Step j ($j \leq m$): Using Statement 1 to find the amount of mix j (T_j) and apply the following formula:

$$\forall i = 1 \text{ to } n+m-j, \mathbf{A}_{i,m-j+1} = T_j \frac{\mathbf{P}_{i,m-j+1,m}}{100} \tag{5}$$

For example, j=4 (the first column): Target amount of Mix 1 is 280=30+50+200 (1,1) = (10/100) *280 = 28; (2,1) = (20/100) *280 = 56; (3,1) = (0/100) *280 = 0; (4,1) = (0/100) *280 = 0; (5,1) = (70/100) *280 = 196;

Basically, calculation of current mix is done by finding target amount of the mix (using Statement 1) and proportions of the ingredients. In order to find the exact amount of each raw material needed to start the technological process, we can just add the amounts of each raw material used in each mix. For example, for Raw Material 1, it would be 78=28+0+50+0.

It should also be noted that using negative amounts does not affect this algorithm since we never assumed that A_{ij} is a positive number in the first place. This represents the fact that some of the raw materials can be various types of water and we can lose water on any technological step.

3 Conclusion

The linear algorithm above answers the question of how much of each ingredient must be used on each step to produce specific amount of target product and gives the exact amount of each raw material needed to start the technological process.

References

[1] Orlicky, J. A. *Material Requirements Planning.* New York, McGraw-Hill, 1971.

[2] Kantorovich, L. V. *A new method of solving some classes of extremal problems.* Doklady Akad Sci SSSR. 28: 211–214, 1940.

[3] G.B Dantzig *Maximization of a linear function of variables subject to linear inequalities.* Published pp. 339–347 in T.C. Koopmans (ed.):Activity Analysis of Production and Allocation, New York-London 1951 (Wiley & Chapman-Hall)

[4] Klee, Victor; Minty, George J *How good is the simplex algorithm?.* New York-London: Academic Press. pp. 159–175, 1972.

[5] Leonid Khachiyan *A Polynomial Algorithm for Linear Programming.* Doklady Akademii Nauk SSSR. 224 (5): 1093–1096., 1979.

[6] Narendra Karmarkar *A New Polynomial Time Algorithm for Linear Programming.* Combinatorica, Vol 4, nr. 4, p. 373–395, 1984.