

Are Your Manufacturing and Distribution Systems Optimal?

Exploiting Networks to Solve Linear Programming Models Faster

Maria G. Lupetin, Lupetin Consultants, Inc.

Trevor D. Kearney, SAS Institute, Inc.

Introduction

This paper will give a brief description of linear programming (LP) and its applications. It will discuss how to look for an imbedded network model with side constraints within an LP model. Models formulated as network models are solved several to hundreds times faster and with less memory than models using the traditional LP solution methods. The capabilities and enhancements of the SAS/OR¹ PROC NETFLOW network solver will be discussed.

Linear Programming Revisited

Linear programming is a mathematical modeling technique which seeks to minimize or maximize a goal. The goal is stated mathematically as a linear equation called the objective function that has the form $c_1x_1 + c_2x_2 + \dots + c_nx_n = \text{total cost or total profit}$. The c 's are coefficients which represent the cost or profit per unit for each variable. The x 's are the decision variables.

In addition to the objective function, the model consists of a system of linear equations or inequalities or both. These constraints mathematically describe conditions that must be met or relationships that must exist for a solution to be feasible. Models may limit the range of values assigned to a variable. If no bounds are specified then the x variable will be assigned a nonnegative value. Optimization software is used to find the values of the x variables, an optimal solution, so that the total cost is as small or total profit is as large as possible subject to the specified constraints.

The LP technique has been used for over four decades to find the "best" solution to applications in areas such as:

- manufacturing and distribution
- transportation and scheduling
- blending/separating of ingredients or chemicals
- military tactics
- financial applications

Typical objectives of linear programming models are to:

- Maximize profits from the sale of products or the charge for services
- Ensure demands are met
- Maximize production by identifying bottlenecks
- Reduce purchase costs of raw materials
- Reduce inventory holding or backorder costs
- Determine the location of facilities and their capacities
- Determine the assignment of resources
- Determine the optimal distribution of goods or services
- Find the shortest path from one location to another

¹SAS and SAS/OR are registered trademarks of SAS Institute, Inc. USA.

The following is an example of a business goal of a manufacturing and distribution firm that can be modeled and solved using the LP technique.

A box manufacturer makes red and blue boxes. Plant A can make a total of 1000 boxes per week, of which 700 can be red and 600 blue in that week. Plant B can make a total of 800 boxes a week. These can be any combination of red or blue boxes. The production costs of producing each colour of box at each factory is known. Customer C requires 200 red boxes and 300 blue boxes per week. Customer D requires 800 red boxes. The transportation costs of each colour of box from each factory to each customer, and the sales price (a profit) charged to each customer for the different boxes are known. The box manufacturer wishes to make and deliver its products at the highest profit possible.

This simple problem could be solved manually; however most real world problems require the use of software packages to generate the constraints and find the optimal solution. "Realistic" linear programming applications can vary in size from a few hundred variables to millions of variables to describe a business need.

At times, linear programming models can be difficult to formulate, solve, validate, explain to management and maintain. It is sometimes difficult to mathematically express all the aspects of reality in a model, and when this is attempted, the LP model usually looks like a formidable algebraic mass of equations, inequalities and bounds. It is also a challenge to confirm that a large model is capturing the business situation properly. To make matter worse, presenting a complex linear programming model to a nonmathematically inclined user can be a trial.

A great deal of human and computer resources can be needed to collect data, set the problem up, optimize and analyze the results. The cumbersome matrix representation of a traditional linear programming model can make the problem set-up and model analysis tedious. Optimization of a large problem can take hours on a mainframe to days on a personal computer. No need for despair. Network formulations and solution methods can reduce many of the difficulties encountered in linear programming as discuss in the next sections.

Networks within Linear Programming Models - Identification

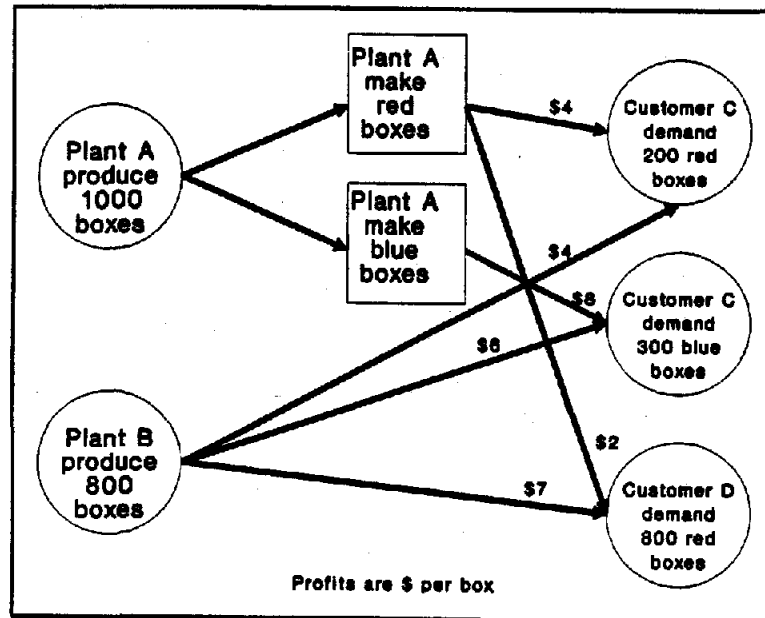
Many LP models, such as the box manufacturer problem, can be depicted by a network. The variables that describe this problem can be drawn as flows across "arcs." The representation of an equation can be drawn as the arcs directed toward and/or away from a "node." Some nodes supply flow; others demand flow. Otherwise, the amount of flow that is conveyed through arcs directed toward the node is equal to the amount of flow through arcs that are directed from that node. The right hand side value of the constraints are represented by the supply (a positive value), or demand (a negative value), or zero (if the node is neither a supply or demand node).

One way to discover an underlying network structure is to inspect the LP constraints. One is looking to isolate the network-type or flow conservation constraints. Consider the coloured box problem, each variable has one 1 and one -1 coefficient in the constraints.² Sometimes the coefficients must be scaled in a constraint (multiplied by some factor) to achieve this structure of 1 and -1 coefficients. If an LP problem has many variables and constraints with 1 and -1 coefficients, the LP model may be algebraically describing a network.

Figure 1 is a representation of the box manufacturer example in network form.

²It is assumed the reader has written the LP model of the box problem for himself.

Figure 1. The Coloured Box Network Model



Frequently, an underlying network structure can be found in large or complicated LP models, especially production-inventory-distribution problems. Drawing the model as a flow of resources from one decision point or node to another, either in space, time or some division or phase in a business operation is one way of finding the embedded network structure. It is not unusual for the modeler to sketch an LP problem as a network diagram when initially formulating the problem. The network diagram is a useful and easy way to understand the structure of a model. Users can comprehend the model without the burden of any mathematical notation.

Networks within Linear Programming Models - Side Constraints

Most LP models cannot be completely described by a network formulation, i.e. "a pure network." When two or more variables must be related to each other to satisfy a constraint and that constraint cannot be depicted as a node, this model is not a pure network. The non-network linear constraints are called "side constraints" since they are part of the formulation but cannot be represented by the network.

Side constraints are used in yield or blending problems where the relationship between two variables, or the relationship between the flows in two arcs must maintain a specific ratio or range of ratios. For example, in a model of the milling industry, the flow through the incoming arcs represent quantities of wheat. After the wheat is processed, the flow through the following arcs are flour and bran. These side constraints model the relationship between the amount of flour and bran produced as a proportion of the amount of wheat milled.

Another type of side constraint is when a group of arcs that are not connected through the same node must be summed to represent a limitation. There may be capacities on transportation or some other shared resource, or limits on overall production or demand in multi-commodity, multi-divisional or multi-period problems. Each commodity, division or period may have a separate network coupled to one main system by the side constraints. Side constraints are used to combine the outputs of subdivisions of a problem to meet overall demands, or to limit overall production or expenditures. In the box manufacturer example, if the firm must store all its products in a warehouse each week, the sum of all the arcs that represent the boxes stored

in the warehouse must be less than or equal to the warehouse capacity. If this stipulation cannot be captured in the network by assigning capacities (upper bounds of the amount of flow that can be conveyed by arcs), a side constraint is necessary.

The Advantages of a Network Formulation and Network Solvers

Network algorithms can solve network programming models several times faster than traditional LP codes solving the equivalent LP models. When LP models grow to tens or hundreds of thousands of variables, they can take hours of solution time on a mainframe or workstation computer. Yet a pure network model using a network solver can reach optimality up to 200 times faster than the same model solved with traditional LP solvers. A network solver can make the difference between providing a solution in a reasonable period of time and within the capacity of the computer, or being faced with an unsolvable model due to a lack of computing resources.

In addition, fast turnaround time is imperative when analyzing many business scenarios using models. As model sizes increase, the computer solution times and memory requirements used by optimizers increase, but fortunately for network packages, these solution time increases are usually much more gradual than when using an LP package.

The nodal flow conservation constraints do not need to be explicitly specified in the network model. They are implied by the network structure. However, flow conservation constraints do make up the data for the equivalent LP model. If you have a LP that is small after the flow conservation constraints are removed, this problem is a definite candidate for solution by a network algorithm. Another benefit of formulating a model as a network rather than a LP is it can more readily be maintained and updated. If the model has to be expanded to take into account other aspects of the business operation, or parts of the model need to have more detail, additional arcs and side constraints can be readily appended to the data files. It is much easier to elaborate a critical part of a network model than change matrices and tables that make up a traditionally formatted LP data.

Overview of PROC NETFLOW

PROC NETFLOW in the SAS/OR software product finds optimal solutions to network programming problems with side constraints. Its use is documented in the SAS/OR User's Guide, Version 6. Additional information for PROC NETFLOW is "A Tutorial on the NETFLOW Procedure in SAS/OR" by Trevor Kearney, in the SAS Users Group International Conference Proceedings, 1990.

PROC NETFLOW's optimizer has been specialized to find optimal solutions quickly. It uses the Primal Simplex Network Algorithm and the Primal Partitioning Algorithm to solve constrained network problems. These algorithms are fast because they take advantage of properties of the network component of the problem. PROC NETFLOW is interactive. It allows one to stop the procedure either on the spur of the moment or when some event occurs, tune the optimizer, save intermediate results, print parts or all of the solution at hand, then resume optimization. The data that PROC NETFLOW requires, and the way solution information is output from PROC NETFLOW is in a form related to a network. It is easier to interpret than the input required for, and output produced by LP codes. The following example is the use of PROC NETFLOW to set up and solve the box manufacturer model.

SAS code for the Coloured Box Problem.

```
/* This dataset contains arc information.          */
/* Each observation has the node where the arc originates, */
/* the node the arc is directed toward,          */
/* the arc's per flow unit (number of boxes per week) profit */
/* (we're maximizing) and the arc's capacity.      */
data arcdata;
    input _from_&$12. _to_&$12. _cost_ _capac_;
    cards;
Plant_A      Plant_A_red      . 700
```

```

Plant_A      Plant_A_blue . 600
Plant_A_red  Cust_C_red   4 .
Plant_A_red  Cust_D_red   2 .
Plant_A_blue Cust_C_blue   8 .
Plant_B      Cust_C_red   4 .
Plant_B      Cust_C_blue   6 .
Plant_B      Cust_D_red   7 .
;
/* This dataset contains nodal supply and demand information. */
/* A positive value is supply, negative value is demand. */
data nodedata;
  input _node_&$12. _supdem_;
  cards;
Plant_A      1000
Plant_B      800
Cust_C_red  -200
Cust_C_blue -300
Cust_D_red  -800
;
proc netflow
  arcdata=arcdata nodedata=nodedata /* input data */
  maximize
  arcout=solution; /* optimal solution output */
run;

proc print;sum _fcost_;run; /* print optimal solution */

```

Messages issued by PROC NETFLOW.

NOTE: Total supply= 1800 , total demand= 1300 .
 NOTE: Number of arcs= 8 .
 NOTE: Number of iterations performed (neglecting any constraints)= 1 .
 NOTE: Of these, 0 were degenerate.
 NOTE: Optimum (neglecting any constraints) found.
 NOTE: Maximal total cost= 8800 .
 NOTE: The data set WORK.SOLUTION has 8 observations and 13 variables.

The SAS dataset called SOLUTION created by PROC NETFLOW contains the optimal solution.

The Coloured Box Optimal Solution.

OBS	_FROM_	_TO_	_COST_	_CAPAC_	_LO_	_SUPPLY_	_DEMAND_
1	Plant_A_blue	Cust_C_blue	8	99999999	0	.	300
2	Plant_B	Cust_C_blue	6	99999999	0	800	300
3	Plant_A_red	Cust_C_red	4	99999999	0	.	200
4	Plant_B	Cust_C_red	4	99999999	0	800	200
5	Plant_A_red	Cust_D_red	2	99999999	0	.	800
6	Plant_B	Cust_D_red	7	99999999	0	800	800
7	Plant_A	Plant_A_blue	0	600	0	1000	.
8	Plant_A	Plant_A_red	0	700	0	1000	.

OBS	_FLOW_	_FCOST_	_RCOST_	_ANUMB_	_TNUMB_	_STATUS_	
1	300	2400	.	7	3	KEY_ARC	BASIC
2	0	0	-2	8	7	LOWERBD	NONBASIC
3	200	800	.	3	2	KEY_ARC	BASIC
4	0	0	0	4	7	LOWERBD	NONBASIC
5	0	0	-5	5	2	LOWERBD	NONBASIC
6	800	5600	.	6	7	KEY_ARC	BASIC
7	300	0	.	2	1	KEY_ARC	BASIC
8	200	0	.	1	1	KEY_ARC	BASIC

=====

8800

Further Enhancements of PROC NETFLOW

PROC NETFLOW is continually being refined to make it execute faster, solve larger problems and use less memory. For example, PROC NETFLOW has solved a constrained network problem of 19,000 arcs and 30 side constraints in .5 minutes of CPU on a mainframe computer. A pure network model of 100,000+ arcs took only eight minutes on a mainframe, (the entire runtime of the procedure included data read and solution output.)

One of the new features of PROC NETFLOW is the option to use IBM's OSL³ to do some or all the optimization. The Optimization Subroutine Library (OSL) is a collection of FORTRAN mathematical subroutines available on IBM mainframes and IBM workstations. OSL provides the use of alternate solution algorithms such as the interior point barrier method. This enables the modeler to select the most efficient set of solution techniques.

PROC NETFLOW can convert a side constrained network problem to the equivalent LP and invoke OSL when specified by an option. If OSL is unavailable or unable to run on the user's computer, PROC NETFLOW will use its own optimizer. PROC NETFLOW can do part of the optimization, and if convergence to the optimum slows, it can have OSL to do any remaining optimization. When the network part of the problem is large compared to the non-network part, especially if the number of side constraints is small, the PROC NETFLOW optimizer is often more efficient than OSL.

One can control OSL in a manner similar to controlling the NETFLOW optimizer. The special options and control parameters needed for the OSL optimizer can be specified in the same format as any SAS procedure's options or parameters. The user need not know or work in FORTRAN, or be concerned with calling the OSL subroutines in a program. The contents of any required input SAS datasets, the output of PRINT statements and the contents of output SAS datasets that have information on the solution are identical no matter which optimizer is used.

Conclusions

Linear and network programming are valuable optimization techniques that can be used to find optimal solutions to a broad spectrum of business problems. Manufacturing and distribution problems are especially amenable to being solved by LP and network programming. Determining optimal solutions has resulted in major savings and improved service for many firms over the decades. PROC NETFLOW is an excellent tool to generate input into a model, solve the model quickly, and report the results.

The representation of a LP model as a network model with side constraints enables one to

- easily define and depict the model
- easily understand and setup the inputs to the model
- solve the model multiple times faster with a network solver than a traditional LP solver
- solve the model with less computer memory
- readily maintain and enhance the model

Authors

Maria G. Lupetin, President
Lupetin Consultants, Inc.
5351 West Winona Street
Chicago, IL 60630 U.S.A.
(312) 736-4059

Trevor D. Kearney
Operations Research Dept.
SAS Institute Inc.
Cary, NC 27513-2424 U.S.A.
(919) 677-8000

³OSL and IBM are registered trademarks of the International Business Machines Corporation, U.S.A.