Efficient Management of Real-World Constraints in a Large-Scale Optimization Model:

An Example of IT-Operations Intervention

Neeraja Y[1,2], Sandeep Lagishetti[1], Sarika K, Sridhar Vallala

IGSA Labs, Hyderabad (www.igsalabs.com)

Keywords: Constraints management, optimization model, IT-operations interface.

Abstract

This paper describes an approach to managing real-world constraints, including a Constraint Engine, which has been developed to aid a large scale optimization model labeled as Manufacturing and Logistics Planner (MLP). MLP has been applied for tactical supply chain planning in several large manufacturing and distribution firms in India. To capture the practical business and operational requirements, the logistics, production and sales departments of a typical large manufacturing company specify numerous constraints. Processing, validating and converting them into mathematical programming manually is a huge time taking process. We overcame this problem by developing a systematic constraint management approach. One part of the approach is the Constraint Engine which is developed as a module within the decision support system (DSS). This improvement process entailed a two-pronged approach which included categorizing constraints into finite generic types and automating the constraint generating and validating process using certain database techniques. The second part is a logic developed to diagnose conflicting constraints in case of infeasibility. This approach drastically cuts down the time taken to process input constraints data, validating them and generating model, thereby permitting quick 'what-if' type scenario analysis.

1. Introduction

This paper describes the Constraint Engine which has been developed to aid the optimization model – Manufacturing and Logistics Planner (MLP). The need for the Constraint Engine arose while implementing MLP in case of tactical supply chain planning for several large firms in

[1] Presenting authors; Presented in SOM Conference 2011 at NITIE, Mumbai.

[2] Corresponding author: neeraja.yellapragada@igsalabs.com

India. These firms typically have 5-20 plants, 50-300 warehouses, and 400-4000 distribution centers (markets). MLP is being used for periodic (weekly or monthly) production and logistics planning.

Constraint: The basic definition of a constraint is a condition that a solution to an optimization problem must satisfy. A business plan needs to be realistic, so it is important to set out in detail the constraints that are likely to act as limits on business activity.

Typical constraints facing a business include:
1.The nature of demand in the market. It is important to identify the nature of customers and their requirements through detailed market research.
2. The nature of the competition. The strength of the competition is a key constraint on business success. Businesses need to position themselves in such a way as to do well in the face of competition.

To capture the practical business and operational requirements, the logistics, production and sales functions of a typical large firm specify numerous constraints – upper, lower and fixed bounds on the production, transportation, mode usage and availability, demand fulfillment, advance allocation commitments, strategic priorities, etc. Some of the constraints change from month to month. The constraints are communicated in normal business language which needs to be converted to mathematical programming language which is understandable to the solver (in this case we use GAMS-CPLEX).

The number of specific types of constraints communicated every month has been in the range of 10-400. Processing, validating and converting them into mathematical programming language manually is a time taking process. We overcame this problem by developing a Constraint Engine as part of the DSS. This improvement process entailed a two-pronged approach which is briefly described below.

The process included:

- Categorizing constraints into finite generic types
- Automating the constraint generating process using certain database techniques

- Diagnosing and resolving the constraints that cause infeasibility (this last step is semi automated)

These improvement efforts culminated in the Constraint Engine that can flexibly generate and validate constraints. This increased the efficiency of the model performance by drastically cutting the time taken to process input constraints data, validate it and generate model.

2. Methodology of Constraint Management

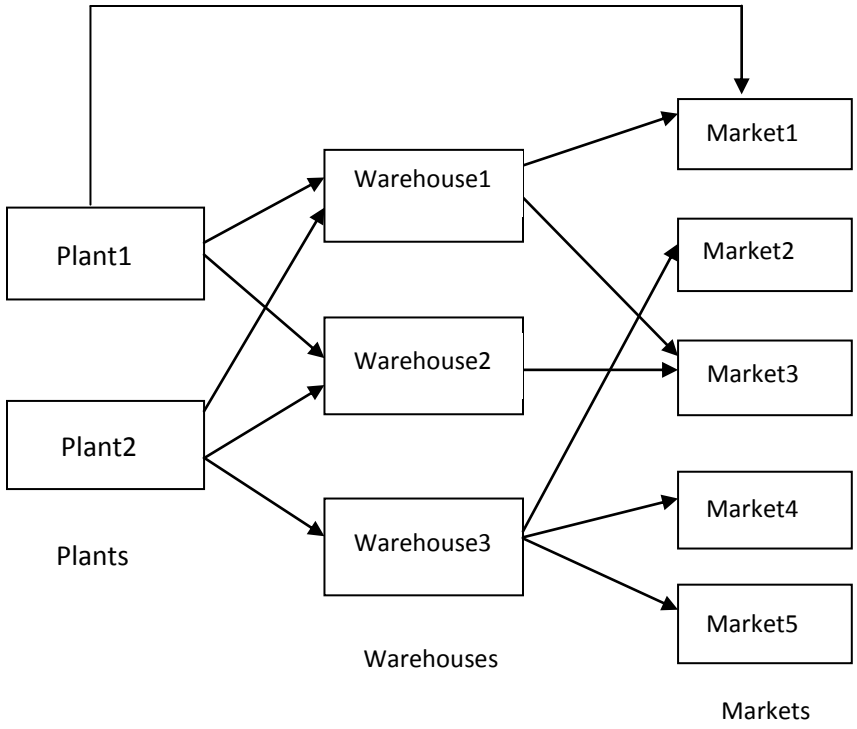

Fig 1. A typical outbound logistics chain

The flow of goods/products from plant to market through warehouses is termed ‘stock transfer movement’ and directly from plant to market is termed ‘direct movement’.

2.1. Categorizing Constraints into Generic Types

In this step we classified all constraints into several generic types, such as Plant to Warehouse movement, Warehouse to Market movement, Plant to Warehouse movement with a given Transport Mode, Outbound movement from a Plant with a given Transport Mode, Direct Plant to

Market movement, etc. Each category of constraints required a common way of writing them mathematically. That is, categories were made so that the mathematical nature of constraint equation (or inequality) was homogeneous within a category and heterogeneous across different categories.

Examples:

i) Plant to warehouse constraints:

Plant Code	Warehouse Code	Transport Mode	Bound Type	Value Type	Value
Plant1	Warehouse1	Rail	Lower Bound	Percentage	20
Plant2	Warehouse3	Road	Fixed	Constant	1000

Explanation to the above constraints:

- Greater than or equal to (at least) 20 % of Plant1's supply must be allocated to Warehouse1 via transportation mode Rail.
- 1000 units (quantity) of Plant2's supply must be allocated to Warehouse3 via transportation mode Road.

ii) Warehouse to market constraints:

Warehouse Code	Market Code	Bound Type	Value Type	Value
Warehouse1	Market4	Upper Bound	Percentage	90
Warehouse2	Market2	Fixed	Constant	450

Explanation to the above constraints:

- Less than or equal to (at most) 90% of the demand in Market4 can be supplied from Warehouse1.

- 450 units (quantity) of the demand in Market2 must be supplied from Warehouse2.

iii) Plant to market with specified product and transport mode constraints:

Plant Code	Market Code	Product code	Transport Mode	Bound Type	Value Type	Value
Plant1	Market1	Prod1	Rail	Upper Bound	Constant	2700
Plant2	Market2	Prod3	Road	Fixed	Constant	5500

Explanation to the above constraints:

- Less than or equal to (at most) 2700 units of demand of the product/SKU Prod1 in Market1 can be allocated from Plant1 via transportation mode Rail.
- 5500 units of Demand of the product/SKU Prod3 in Market2 must be allocated from Plant2 via transportation mode Road.

iv) Plant with transport mode constraint:

Plant Code	Transport Mode	Bound Type	Value Type	Value
Plant2	Rail	Upper Bound	Constant	4500
Plant6	Road	Fixed	Percentage	45

Explanation to the above constraints:

- 4500 units of supply from Plant2 must be via transportation mode Rail.
- 45% of supply from Plant6 must be via transportation mode Road.

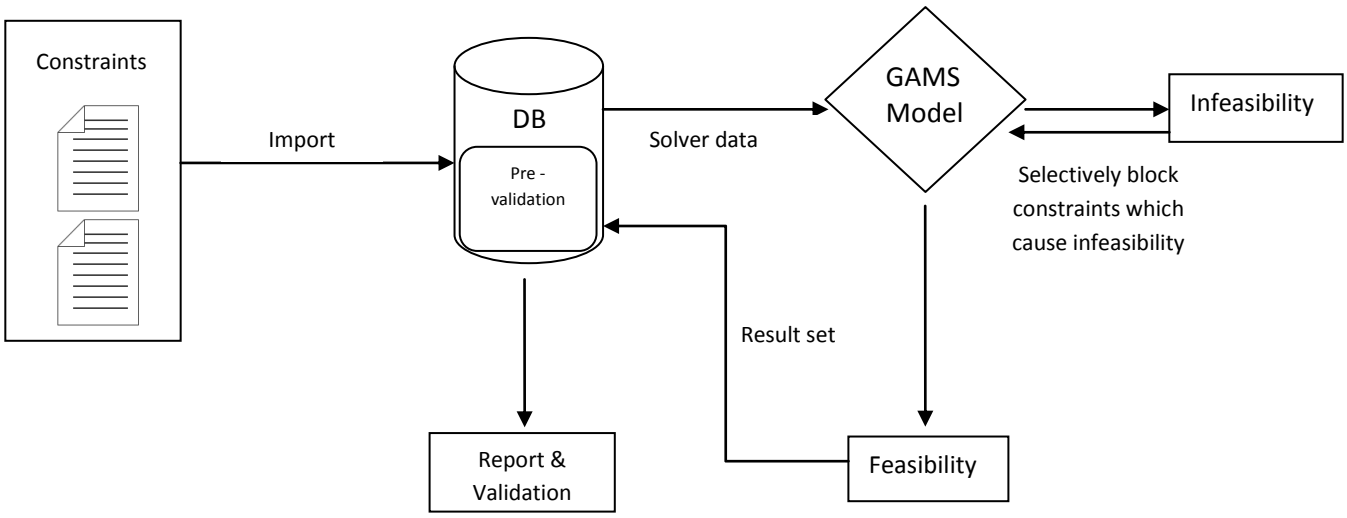

Fig 2. Methodology of Constraint Engine and DSS

2.2. Automating constraints generation process using database techniques

a) Based on the constraints categorization we prepared standard input templates to input the constraints data into the database.

Example of Input Template format:

Plant Code	**Transport Mode**	**Bound Type**	**Value Type**	**Constraint Type id**	**Value**
Plant2	Rail	Upper Bound	Constant	CT4	4500
Plant6	Road	Fixed	Percentage	CT4	45

The constraint type id is used to differentiate each category of the constraint.

Based on the constraint type id, the constraint engine generates mathematical equations (or inequalities) for different constraint types.

b) After importing the constraints data into the database we programmed the required validation SQL queries. These queries are useful to validate the initial feasibility of each constraint. For instance, a constraint specified for a particular Plant-Warehouse

movement is immediately identified if there is no corresponding logistics link or no such master data exists.

c) After constraints validation process, the model is generated with all the constraints that have passed initial validation.

   An example of constraints that are generated in the form of Mathematical programming language in GAMS model using CPLEX solver:

   Plant2_Rail(t)..sum((p,j,m)$set_rail(m),Y(p,"Plant2",j,m,t))=l=4500;

   Explanation to the above constraint:

   Indices: p: Product, j: Warehouse, m: Transportation Mode, i: Plant, t: Time-Period

   Set: set_rail(m) it is a set containing only the rail links from set m

   Variables: Y: Plant to Warehouse movement

   Constraint name: Plant2_Rail (this is applicable for every time-period)

d) The next automated step executes the optimization model-solver.

e) If the model run shows a feasible and optimal result, then the raw results are imported into the database. In the next step we validate the results by using further database SQL queries for consistency of production and logistics flows (for ex. we check if the total flows match and if the lower bound constraints are satisfied and to what extent). After validating the results, we generate output reports in Excel.

f) If the model run shows infeasibility, then we diagnose those particular constraints which cause infeasibility and block them iteratively and take several runs until the model run shows a feasible solution. In this iterative process we often communicate back-and-forth with the client firm and clarify/modify the bounds of the constraints that cause infeasibility. This process goes on until a feasible and optimal solution to the satisfaction of the client firm is obtained. Then the results are imported into the database. Then we validate the results by further database SQL queries for consistency of flows and generate reports in Excel.

Handling Infeasibility:

In order to diagnose the constraints that cause infeasibility, we enable different types of constraints in the markets-to-plants sequence, i.e. from downstream to upstream.

- From the different categories of constraints we first retain and execute only market based constraints in order to check if the plants can meet all market demands without any intermediate constraints on warehouses or transport modes.
- Next, we add the transport mode constraints at the level of plants in order to check feasibility of mode availability at specific plants.
- Next, we execute plant wise, plant to warehouse allocation constraints.

In such a systematic way we are able quickly find out which constraints are causing infeasibility. Then we block those constraints or modify their right-hand-side bounds after consulting the client firm, and re-run the model until the run shows feasibility.
All these iterations of diagnosing infeasibility are done manually.

Reasons for Infeasibility could be any of the following:

- Insufficient plant capacities.
- Insufficient transport links (routes) from plants to markets, either directly or via warehouses.
- Incomplete or incorrect input data. Several Indian companies do not properly maintain complete or correctly updated data in ERP system, and this could be a reason for infeasibility.
- Certain constraints may conflict with each other.

3. Conclusions:

- The Constraint Engine automates the entire process of processing constraints data, validating and generating corresponding equations and identifying constraint level causes for infeasible solutions.
- In this process we have saved a lot of time. In typical scenarios for a large firm these steps were taking more than a full day, whereas now it takes only an hour or less.
- This way the user can obtain quick results and construct and run several ‘what-if’ type scenarios with the model, and take more informed decisions.