

**An Interactive Business Intelligence
Platform for Generic Optimization
Based DSS : Illustrated with a Supply
Chain Management Problem**



College of Engineering, Design and Physical Sciences
Brunel University London
United Kingdom

A thesis presented for the degree of
Master of Philosophy

by

Ansuman Swain

Department of Mathematics

October 2018

Abstract

The aim of this thesis is to define an approach to create an interactive optimization based Decision Support System (DSS) with Business Intelligence (BI) capability. The decision process involves the formulation of a problem and finding an optimal solution. The data and results associated with the model can be retrieved and analysed further to make important business decisions. We expand on this aspect of a generic framework which can be used for creating a dashboard to control the underlying model. Since decision models involve uncertainty, practical implementation of such a model includes a large number of constraints, datasets and scenarios. If the formulation of the model and its abstraction is sufficiently generic, then the analysis of the data and results can be also achieved using the interactive dashboard. This dashboard is web-based and can be operated by all DSS users without any expertise in optimization system components.

In terms of mathematical programming, the decision models under investigation are linear programming problems. The formulation of the models are implemented in A Mathematical Programming Language (AMPL), which is a well-established Algebraic Modeling Language (AML) for Optimization. The input and output from the modeling system are summarized into multidimensional data views for Online Analytical Processing (OLAP). The unique feature of the framework includes the instantiation of a model with different datasets from an interactive web-based platform. These datasets and results can be analysed and visualized further with the help of a BI tool. The generic use of the framework is presented with optimization problems from diverse domains such as Supply Chain Management and Asset Liability Management.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Cormac Lucas for his continuous guidance and support throughout the year.

My deepest thanks to my industrial supervisor Prof. Gautam Mitra for his invaluable advice and ideas which helped in the accomplishment of the research work.

I am extremely thankful to my financial sponsor OptiRisk Systems for providing a supportive platform. All the staff members were always supportive and helped me shape up my research skills.

I would also like to acknowledge the academic staffs of the Department of Mathematics at Brunel University for their help and cooperation.

Last but not least, I also thank my parents and my wife Shradha for always being there and encouraged me throughout. I could not have done it without them.

List of Figures

2.1	A Supply Chain Management System	10
3.1	Interaction Based on Roles	24
4.1	System Architecture	27
4.2	Model Sets	30
4.3	Parameters	31
4.4	Indexed Parameter	31
4.5	Solver Panel Success	32
4.6	Solver Panel Error	32
4.7	Supply Chain Dashboard	36
5.1	Structure of Relational Table	38
5.2	OLAP Cube from Data	40
5.3	OLAP Cube <i>Dealerdemand</i>	41
5.4	Aggregated OLAP Cube	47
5.5	Cube <i>Dealerdemand</i> with Sliced Data	49
5.6	Cube <i>Dealerdemand</i> with Diced Data	51
6.1	Available Cubes	54
6.2	Aggregated Data Table for <i>initInv</i>	55
6.3	Dimension for Data Table <i>initInv</i>	56
6.4	Detailed Data Table <i>initInv</i>	56
6.5	Filtered Data Table	57
6.6	Chart Options	58
6.7	Supply from Factory to Warehouse [Aggregated]	59
6.8	Supply from Factory to Warehouse [Drilled Down]	60
6.9	Supply from Factory to Warehouse [Drilled Down with filter]	61
6.10	Production in Factories over Time Period	62
6.11	Production Capacity at Factories	62

6.12	BI System as Feedback Loop	65
7.1	ALM Dashboard	72
7.2	Interactive Dashboard	74
7.3	<i>MarketValue</i> Data Drill Down over Time	75
7.4	<i>MarketValue</i> Bar Chart Drill Down over Time	76
7.5	<i>MarketValue</i> Pie Chart Drill Down over Time	77
7.6	A Customized Version of ALM Dashboard	78
A.1	Configuration File	96

List of Tables

2.1	Model Sets	11
2.2	Model Parameters	12
2.3	Model Decision Variables	13
3.1	Sets of the Model	19
3.2	Production Cost and Capacity	20
5.1	Drill Down across One Dimension [Product]	45
5.2	Drill Down across Two Dimensions	45
7.1	List of Sets	67
7.2	List of Parameters	68
7.3	List of Variables	68
A.1	Inventory Details	93
A.2	Warehouse Details	93
A.3	Transportation Cost (Warehouse to Dealer)	94
A.4	Transportation Cost (Factory to Dealer)	94
A.5	Transportation Cost to Warehouse	94
A.6	Shortfall Penalty	95
A.7	Dealer Demand	95
A.8	Asset Sector Details	97
A.9	Data Table for Assets	97
A.10	Asset Price	98

Acronyms

ALM Asset Liability Management

AML Algebraic Modeling Language

AMPL A Mathematical Programming Language

API Application Programming Interface

BI Business Intelligence

DSS Decision Support System

JSON JavaScript Object Notation

MDDB Multidimensional Database

OLAP Online Analytical Processing

Contents

Abstract	ii
Acknowledgement	iii
List of Figures	v
List of Tables	vi
Acronyms	1
1 Introduction	4
1.1 Background and Context	4
1.2 Aim and Objectives	5
1.3 Organisation of Thesis	6
2 Decision Problems in DSS	8
2.1 Problem Statement	8
2.2 Algebraic Model Components	10
2.3 Objective Function	13
2.4 Constraints	14
3 Model Formulation and Results	17
3.1 Formulation of the Model	17
3.2 Further Analysis	21
4 An Interactive DSS Framework	26
4.1 Overview of the System Architecture	26
4.2 An Interactive Dashboard for the Model	28
4.3 Instantiation with Master Dataset	29

4.4	Dashboard Controls of Model Components	30
5	Model Data Analysis	37
5.1	Evolution of Relational DBMS	37
5.2	Multidimensional Databases	39
5.3	OLAP Features Illustrated with Model Data	41
6	Data Visualization and Business Intelligence	52
6.1	A Visualization System for OLAP	52
6.2	Data Views (Tables)	54
6.3	Graphs and Charts	57
6.4	A Perspective from a Business Owner	61
6.5	Summary of BI platform	65
7	Use of the Framework in other Domains	66
7.1	The ALM Problem	66
7.2	Algebraic Model	67
7.3	Model Formulation and Result	70
7.4	Dashboard for the Optimization Model	71
7.5	Further Analysis	73
7.6	Analyst BI Platform	75
8	Conclusion	80
8.1	Overview	80
8.2	Summary	80
8.3	Research Aim and Objectives	82
8.4	Contributions	83
8.5	Limitations and Future Directions	84
8.6	Concluding Remarks	85

Appendix A Illustrative Data and Configurations	93
A.1 Input Data Tables	93
A.2 Configuration file for Dashboard	96
A.3 Data Tables : Asset Liability Management	97
Appendix B AMPL Files	99
B.1 Model File : Supply Chain Management	99
B.2 Data File : Supply Chain Management	105
Appendix C AMPL Files	109
C.1 Model File : Asset Liability Management	109
C.2 Data File : Asset Liability Management	112
C.3 Script File : Asset Liability Management	112

1 Introduction

1.1 Background and Context

In a broader context, a Decision Support System provides aid to the corporate decision-making process. Over the last few decades, DSS applications have significantly evolved and are almost used in every domain and business. The earlier days of DSSs were focused around the development of the model and problem analysis, but in recent years other powerful tools have played a key role. A broad overview of DSSs and their evolution over the last few decades is given in Shim et al. 2002. It also focuses on typical DSS components like a data warehouse, OLAP, data mining and the user interface. Additional concepts that came into DSSs include Group Decision Support Systems (Gray 1987, Desanctis et al. 1987), artificial intelligence (Whinston et al. 1981) and machine learning (Shaw et al. 1988). These components have become the key focus areas for researchers and corporates. The recent development in web-based technology has also played a critical factor in DSSs. A systematic approach to build a web-based DSS was first introduced by D. Power et al. 2001. Bharati et al. 2004 investigated the factors affecting decision making in web-based decision support systems.

Researchers over the years have proposed several ways to classify a DSS. Classification of a DSS is not simple since it may fit into more than one category. Examples of different types of DSS include data-driven, model-driven, knowledge-driven, document-driven and communications-driven DSS. An optimization-based DSS is one of the important types among the many classes of DSSs and can be considered as a model-based DSS. In optimization based DSS, the model and database experts play a vital role in developing the model and data components. Constant efforts are required

by these experts in different stages of the decision-making process. As the model gets complex, the role gets more critical. Furthermore, problem owner is responsible for taking all the factors into consideration in relation to the change in decision model components. In most cases, the process of result analysis and reporting are also monitored closely by problem owner by the help of the BI platform. The role of knowledge from such BI systems helps in corporate decision making. However, there are a few pitfalls in this type of set up. Consistent intervention from three different parties is not ideal in a quick decision-making process. The BI system provides insight based on available input data alone. BI systems are isolated from the optimization system and specialized connectivity is required to extract data from the optimization system for different business scenarios. A very small number of academic literature tries to link these systems together.

1.2 Aim and Objectives

The aim of this thesis is to provide an interactive BI platform for generic optimization based DSS. This is illustrated with a supply chain management problem.

The following objectives are considered in order to achieve the aim of this thesis:

Objective 1: To set out and understand optimization-based decision problems for defined domains. Furthermore to identify key entities of the decision problem in terms of sets, indices and variables of a constrained optimization system.

Objective 2: To formulate an abstract and data-driven optimization model for the decision problem based on the algebraic model.

Objective 3: To design an interactive web-based dashboard to analyse and solve the optimization model. This dashboard is aimed to provide different instances of the underlying decision problem based on user interactions.

Objective 4: To provide an analysis method for multidimensional input data and results associated with the optimization model.

Objective 5: To integrate a tightly coupled BI system into the web-based dashboard. Also, to ensure the resulting system supports exploiting data from various input sources as well as from the instances of the model created by the dashboard.

Objective 6: To demonstrate the generic nature of the proposed platform with similar optimization-based decision problems from other domains.

1.3 Organisation of Thesis

The remainder of this thesis is organised as follows. Chapter 2 provides a general introduction to decision problems with a detailed example of supply chain problem. Then we define the components of the model and create an algebraic form of an optimization model. Chapter 3 covers the formulation of the model in AMPL. Associated input data and solution methods are also presented. In chapter 4, we introduce the system architecture of an interactive web-based DSS. Various model components and the interaction of the model with a dashboard are also illustrated. In chapter 5, we introduce the generic approach for data analysis associated with the optimization model. OLAP cubes and its resemblance to multi-dimensional data is discussed in this section. Various examples are given for OLAP features. A brief overview of the OLAP framework used is also included. Chapter 6 provides an overview

of a visualization framework for OLAP and its features. Use of a BI platform for the problem owner is also included in this section with a few business scenarios. Chapter 7 shows the application of the framework in another domain. The example used in this case is of an Asset Liability Management (ALM) problem. Finally, a summary of the entire framework and its generic use outside of the specific problem is discussed in section 8. A number of appendices include the optimization model and data files written in AMPL.

2 Decision Problems in DSS

In most cases, DSSs are presented within the context of domain-specific problem-solving (Dengiz et al. 2006, Dutta et al. 2007). This research includes optimization-based DSS for two different decision problems. In this chapter, a supply chain management problem is introduced along with its algebraic formulation. Examples are given using this problem in subsequent chapters. The second problem deals with asset liability management. A complete overview of this is presented in chapter 7.

With the introduction of new technology and concepts, establishing the right methodologies for supply chain management has become vital for many organisations. DSS with supply chain management (Koutsoukis et al. 2000) helps in identifying various strategies for business problems. Some of these strategies are followed by corporates at different points of time as given in Borade et al. 2007. Although the techniques vary from organisation to organisation, the core concept of a supply chain is very basic. In simple terms, it is a process that involves the transfer of materials in order to fulfill the demand at the consumer end. The purpose of this research is to explore the integration of optimization based DSS. Keeping this in mind, a simple form of supply chain network is presented. A similar type of example was proposed by Das et al. 1994 for a wholesaling system. A prototype data-set and forecasted values are used in our case. Furthermore, the constraints and bounds presented are based on natural assumptions.

2.1 Problem Statement

Figure 2.1 shows an example of a supply chain management system. One use case can be a multinational garments manufacturer. There are factories

at different locations, where production and packaging take place. Different factories have different production rates which are known to the problem owner at the start. Some of the factories are equipped with inventory facilities. This helps for storage during the post-production phase without transferring products to separate warehouses. The initial inventory at the start of the time horizon is also known.

The following assumptions are also considered :

- The manufacturing company knows all types of costs associated with the production process. Different costs include production costs, inventory costs and transportation costs between different locations.
- Factories and warehouses can store products according to their respective capacities. This is known as well and assumed constant over the time horizon.
- Keeping the customer's goodwill in mind, a shortfall penalty is applicable in the case of not meeting demand.
- The dealers are served from the warehouse locations. In some cases, the dealers can get their products directly from the factories.

All the above entities are assumed constant over the time horizon. However, customer demand varies over time and can never be forecasted exactly. In this case, we have used the expected dealer demand forecast for each period in the time horizon.

Decisions for the Problem Owner : The decision owner needs to evaluate various cost factor associated with the process. The goal is to minimize the total cost over the time horizon.

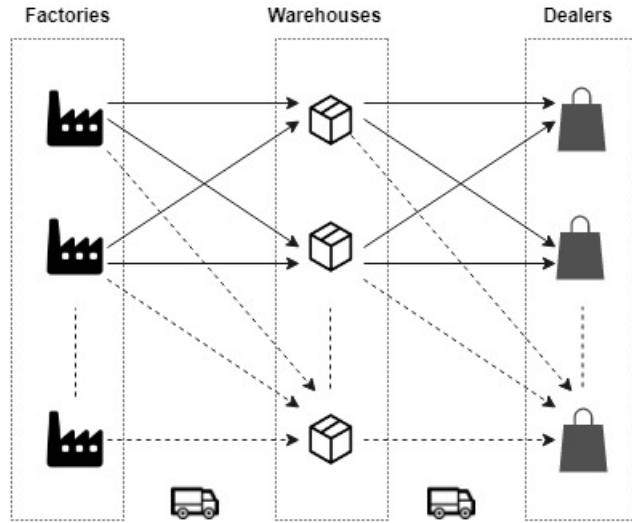


Figure 2.1: A Supply Chain Management System

2.2 Algebraic Model Components

In this section, we introduce the model components progressively. The notations defined are used in the algebraic form of the supply chain problem. A constrained optimization problem consists of an objective function and constraints. These are stated in terms of parameters and decision variables.

2.2.1 Sets and Indices

Garment products can be divided into many sub-categories. For design simplicity, we have grouped all the different products (e.g Shirts, Skirts, Jeans etc.) into a single set. A list of all the sets used in the model is given in Table 2.1.

Notation	Description
P	Set of Products
F	Set of Factories
W	Set of Warehouses
D	Set of Dealers
T	Set of Time periods

Table 2.1: Model Sets

2.2.2 Parameters

From the qualitative description of the problem, we have identified the following parameters (Data). Some of these parameters are indexed over sets given in Table 2.1. A list of all the parameters associated with the supply chain problem is set out in Table 2.2.

Notation	Description
PC_{pf}	Cost of production of product p at factory f
IC_{pf}	Cost of holding product p at factory f
WC_{pw}	Cost of holding product p at warehouse w
PE_{pd}	Penalty for shortfall of product p at dealer d
T_{fw}	Transportation cost from factory f to warehouse w
T'_{fd}	Transportation cost from factory f to dealer d
T''_{wd}	Transportation cost from warehouse w to dealer d
CP_{pf}	Production capacity of product p at factory f
CF_{pf}	Inventory capacity of product p at factory f
CW_{pw}	Inventory capacity of product p at warehouse w
IF_{pf}	Initial holding of product p at factory f
IW_{pw}	Initial holding of product p at warehouse w
DE_{pdt}	Expected Demand for product p at dealer d at time t

Table 2.2: Model Parameters

2.2.3 Decision Variables

Decision variables in a mathematical model define the quantities to be determined by the decision maker. These are set out in Table 2.3.

Notation	Description
Q_{pft}	Production quantity of product p at factory f and time t
q_{pfmt}	Amount of product p at factory f to be supplied to warehouse w at time t
q'_{pfdt}	Amount of product p at factory f to be supplied to dealer d at time t
q''_{pwdt}	Amount of product p at warehouse w to be supplied to dealer d at time t
S_{pdt}	Shortfall of product p at dealer d in time t
Z_{pft}	Amount of product p to be held in inventory at factory f in time t
Z'_{pwt}	Amount of product p to be held in inventory at warehouse w in time t

Table 2.3: Model Decision Variables

2.3 Objective Function

In this case the goal of the decision maker is to minimize the total cost to the manufacturing company. The total cost is the sum of costs incurred due to production, transportation, inventory and shortfall penalty.

$$\begin{aligned}
Cost_{total} = & Cost_{production} + Cost_{transportation} \\
& + Cost_{inventory} + Cost_{penalty}
\end{aligned}$$

This is expressed in equation 1 as shown below.

$$\begin{aligned}
\text{minimize } C_{tot} = & \sum_{p \in P} \sum_{f \in F} \sum_{t \in T} PC_{pf} * Q_{pft} \\
& + \sum_{p \in P} \sum_{f \in F} \sum_{w \in W} \sum_{t \in T} T_{fw} * q_{pfmt} + \sum_{p \in P} \sum_{f \in F} \sum_{d \in D} \sum_{t \in T} T'_{fd} * q'_{pfmt} \\
& + \sum_{p \in P} \sum_{w \in W} \sum_{d \in D} \sum_{t \in T} T''_{wd} * q''_{pwdt} \\
& + \sum_{p \in P} \sum_{f \in F} \sum_{t \in T} CF_{pf} * Z_{pft} + \sum_{p \in P} \sum_{w \in W} \sum_{t \in T} CW_{pw} * Z'_{pwt} \\
& + \sum_{p \in P} \sum_{d \in D} \sum_{t \in T} PE_{pd} * S_{pdt}
\end{aligned} \tag{1}$$

2.4 Constraints

Constraints are restrictions placed on the available resources. We represent different types of constraints for the above problem.

2.4.1 Product Balance Constraint

At any time period, we endeavour to balance the dealer's demand for each product. Apart from the shortfall, the sum of all products received from warehouses, as well as factories, should be equal to the forecasted demand.

$$DE_{pdt} = S_{pdt} + \sum_{f \in F} q'_{pfmt} + \sum_{w \in W} q''_{pwdt} \quad p \in P, d \in D, t \in T \tag{2}$$

2.4.2 Inventory Balance Constraint

As per the problem description, the surplus products get stored in inventory (located at the factories). Thus apart from the stored products, the sum of all the incoming products must be equal to the sum of all the outgoing products. This is applicable for all time periods. Stored products at the end of the

previous time period can be utilized at the next time period for meeting the demand in certain locations. Similarly, the initial inventory storage can be utilised at the beginning of the time horizon. Hence the inventory balance can be expressed at the initial time period and subsequent time intervals as given in equation 3 (at $t = 1$) and equation 4 (from $t = 2$ onwards).

Initial inventory,

$$IF_{pf} + Q_{pf1} = \sum_{w \in W} q_{pfw1} + \sum_{d \in D} q'_{pfd1} + Z_{pf1} \quad p \in P, f \in F \quad (3)$$

Inventory balance in subsequent time periods,

$$Z_{pft-1} + Q_{pft} = \sum_{w \in W} q_{pfmt} + \sum_{d \in D} q'_{pfdt} + Z_{pft} \quad p \in P, f \in F, t \in 2, \dots, T \quad (4)$$

2.4.3 Warehouse Balance Constraint

The warehouses receive products from factories depending on their capacity and transportation cost to dealers. Warehouses can store products until further demand arises at the dealers end. Thus the inventory balance constraint can be applied to the warehouse as well.

Initial warehouse,

$$IW_{pw} + \sum_{f \in F} q_{pfw1} = \sum_{d \in D} q''_{pwd1} + Z'_{pw1} \quad p \in P, w \in W \quad (5)$$

Warehouse balance in subsequent time periods,

$$Z'_{pwt-1} + \sum_{f \in F} q_{pfmt} = \sum_{d \in D} q''_{pwdt} + Z'_{pwt} \quad p \in P, w \in W, t \in 2, \dots, T \quad (6)$$

2.4.4 Bounds

The following bounds are also applicable :

- The production at each factory should not exceed the maximum capacity.

Bound on production capacity,

$$Q_{pft} \leq CP_{pf} \quad p \in P, f \in F, t \in T \quad (7)$$

- The inventory storage should be less than the maximum permissible limit.

Bound on inventory capacity,

$$Z_{pft} \leq CF_{pf} \quad p \in P, f \in F, t \in T \quad (8)$$

- The storage capacity of the warehouse should be less than the maximum permissible limit.

Bound on warehouse capacity,

$$Z'_{pwt} \leq CW_{pw} \quad p \in P, w \in W, t \in T \quad (9)$$

Summary

This chapter describes a sufficiently generic supply chain optimization problem. An equivalent algebraic formulation of the model is presented throughout from equation (1) to (9). These include different components of the model, objective function, decision variable and constraints. Based on this, a detailed framework is proposed in section 4. A similar structure is followed for the asset liability management problem in chapter 7.

3 Model Formulation and Results

In the field of operational research, linear programming plays a key role in addressing optimization problems. It helps in the formulation of a business decision problem in relevant mathematical form. Mainly the objective of any linear programming problem is the maximization of profit or minimization of cost. Since the decision problems under investigation are deterministic and the goal is to find minimum cost or maximum revenue, linear programming is well suited to our purpose. Real-world implementation of a business problem may include large non-linear models due to complexity and uncertainty factors.

This chapter explains the formulation of the business problem using algebraic modeling language tools. These tools help in expressing algebraic equations in the form of abstract modeling entities like sets, indices, parameters, variables and constraints. The formulated model is instantiated with a sample data-set. The instantiated model is solved by the help of an external linear solver. Various analysis methods are discussed which includes input data and results given by the solver.

3.1 Formulation of the Model

There are numerous AMLs which can convert linear and non-linear programming problems to a machine-readable form. Some of the state-of-the-art AML systems are LINDO/LINGO, AMPL, MPL, AIMMS. Kallrath 2004 explained the use of these AMLs in a mathematical optimization problem. In this research, we used AMPL for the formulation of the decision model. AMPL is widely used in industry and academia. The structure and syntax of a model in AMPL resemble very closely to the mathematical form. In

addition to this, AMPL provides a series of solvers and integration to other systems via Application Programming Interface (API). The detailed use of API is given in chapter 4. The concept and framework of this research are applicable to other modeling systems like GAMS and AIMMS as well. More details about the AMPL language and its syntax are available in Fourer et al. 2002.

3.1.1 The AMPL Model

The task of modeling in AMPL can be further subdivided into three parts. i.e. The model, data and solution. It is good practice to separate the model from the data. This gives the flexibility to instantiate the same model with different input datasets. In other words, it makes the system data driven which is ideal when dealing with real-world optimization problems.

- The AMPL model contains the definitions of sets, parameters and variables. Objective functions and constraints are also part of the model file. The mathematical equations are represented in the equivalent AMPL syntax which is very similar to the algebraic notation.
- The AMPL data file contains the values which are to be assigned to the model entities before they are used.
- The solution part includes various AMPL commands which interact with sending the data to different types of solvers and receiving the solution back.

The AMPL model of the supply chain problem is shown in Appendix B. The notations and syntax can be compared easily to the algebraic model given in chapter 2.

3.1.2 Input Data

In this example, we have used a prototype data set. The AMPL model described in the previous section is based on this data set. A data source can be anything like an excel spreadsheet, static data files, AMPL data file, or CSV file or any relational database. A Relational Database Management System (RDBMS) is the most preferred database system due to its simplicity in design (Tables), better retrieval mechanism and multi-user accessibility. Input data for the AMPL model are represented in the tables below. The entire data file contents are given in Appendix B.

From the model definition, we have identified 5 input sets. Values for these sets are shown in Table 3.1

Factory	Warehouse	Product	Dealer	Time
Southall	Norwich	Skirts	London	1
Leeds	Leicester	Shirts	Paris	2
		Jeans	Wien	3
				4

Table 3.1: Sets of the Model

Over the time horizon, cost and capacity of the products remain constant at a factory. The production cost and capacity for the products are shown in Table 3.2. Capacity values present the number of units of product. Cost values are in pounds per unit (£/unit).

As explained in the problem statement, the factories are equipped with inventories to store products and corresponding parameters like the capacity, storage cost and initial inventory holdings are known to the problem owner. These details are given in Table A.1 of Appendix A.

Factory	Product	Production Cost	Capacity
Southall	Skirts	1.5	36
Southall	Shirts	1.5	0
Southall	Jeans	1.5	85
Leeds	Skirts	1.5	54
Leeds	Shirts	1.5	60
Leeds	Jeans	1.5	36

Table 3.2: Production Cost and Capacity

Warehouses also deliver the products to different dealer locations. The parameters like capacity, storage cost and initial warehouse holdings are given in Table A.2 of Appendix A.

Figure 2.1 shows a possible supply chain network discussed in the problem statement. The transportation links include transportation between factories and warehouses as well as between warehouses and dealers. Under certain circumstances, factories can supply directly to dealers in different locations. Tables A.3, A.4, A.5 in Appendix A show the transportation costs between factories, warehouses and dealers.

In the case of demand not being met, there is a penalty applicable for each product and each dealer. This is represented in Table A.6 of Appendix A. For design simplicity, we have assumed that the expected demand is known at the start of the planning horizon. This is given in Table A.7 of Appendix A.

3.1.3 Solution

AMs solve optimization problems with the help of solvers. AMPL comes with a number of linear and non-linear solvers. Once the model is formulated and instantiated with a dataset, the next set of tasks includes solving the

model. Usually, the model instance is sent to the solver where an optimization algorithm is executed and the results are returned. In this problem, the CPLEX solver is adopted which is widely used for solving linear programming problems using the simplex method. CPLEX can easily be hooked into the AMPL system. A detailed technical guide on different solvers supported by AMPL can be found in Gay 1997.

AMPL provides a list of commands to interact with the underlying solver. The results are returned from the solver when we try to solve the model as given in Appendix B.

Solver Output :

```
CPLEX 12.6.0.0: optimal solution;
objective 10270.2146
128 dual simplex iterations (0 in phase I)
costTot = 10270.2
costTot.slack = 10270.2
```

3.2 Further Analysis

Analysis of the resulting output of an optimization system can be difficult at times. However, it is extremely important for the decision-making process. During the analysis phase, the decision maker needs to evaluate different input scenarios and values to determine a range of possible outcomes. This helps in better understanding of the problem as well as better decision making. Here two types of analysis are explained. Data-driven analysis based on different associated input data in different business scenarios. The role-based analysis focuses on the analysis performed by different users in a typical optimization bases DSS.

3.2.1 Data-driven analysis

The result of the model solved with the default dataset is given in section 3.1.3. Generally, a modeling system involves a number of problem instances which differ from each other by means of the input data. The model discussed in this research is a deterministic optimization model. Let us consider a mathematical model which is represented as an LP (Linear programming problem) in matrix form as given in equation (10)

$$\begin{aligned} & \text{minimize } Cx \\ & \text{Subject to } Ax \leq B \\ & \text{and } x \geq 0 \end{aligned} \tag{10}$$

In the above equation, the known coefficients are vectors B and C and the matrix A . We can represent a collection of coefficients as given in (11).

$$D = \{A, B, C\} \tag{11}$$

Let us assume the decision maker wants the objective function and the constraints to be updated. Similarly, the decision maker can decide to relax a limitation proposed in (10). Hence constraint coefficients need to be changed. Given a collection of coefficient data as presented in (11), we can create a different collection of elements. In a generic way the set of coefficients can be written as below

$$d_i = \{A_i, B_i, C_i\} \quad i \in 1, 2..n \tag{12}$$

If the problem consists of a large number of decision variables and constraints, different business scenarios come with a number of different

coefficients. Let us consider d_1, d_2, \dots, d_n as the set of coefficients for the optimization problem for n different scenarios. The coefficient vectors are known to the problem owner from the beginning. Thus it is feasible to create a master list of coefficients with all the possible values. If we assume D as a master set of coefficients from (11), then the relationship between the set of coefficients in individual example scenarios and the master set can be represented as below

$$D = d_1 \cup d_2 \cup d_3 \dots \cup d_n \quad (13)$$

This leads to the idea of creating separate model components and data components. Furthermore, the data components should be created in such a way that the model can be instantiated and configured by a different set of input data for different business conditions. Choosing the master set of coefficient values is also essential as it gives more flexibility and freedom to the problem owner.

3.2.2 Role-based analysis

DSS often include different types of users. Users, based on their roles are responsible for analysing different DSS components. A brief interaction of users analysis based on their roles is given in Figure 3.1. The segregation of users in different parts of the DSS makes analysis more effective.

Model experts are responsible for developing the decision model which is the backbone of the DSS. They also analyse model components and modify them when required. Model investigation at times requires instantiation of the model with a set of input data and evaluation of the output parameters(results). This

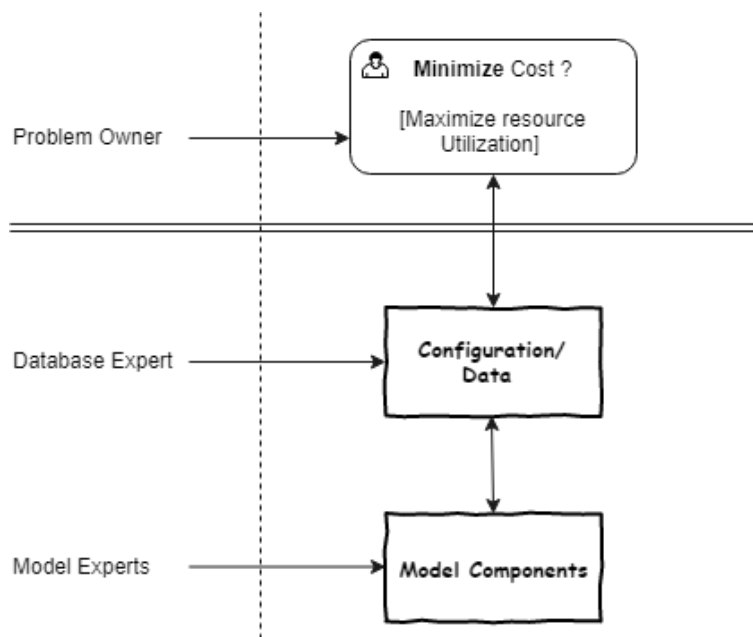


Figure 3.1: Interaction Based on Roles

may lead to changes in the model components in order to achieve alternative decisions.

Database experts are responsible for analysis of the transactional or analytical database. In the case of a DSS, it can be challenging if there are many decision variables which are multidimensional. This case study introduces a novel approach (OLAP Analysis) to the database analysis of the problem. This approach is quite abstract and applicable to any type of optimization problem, which produces a set of multidimensional results. Although the concept of OLAP cube is not new in data warehouse, we lack any system that exploits the data associated with an optimization system for analysis purposes. This research shows the integration of an optimization system and associated data with an OLAP system and analyst tools (BI reporting).

Problem owners in a corporate organisation are responsible for analysing the problem and taking business decisions. Keeping this in mind, we have introduced an easy-to-use dashboard. A generic framework is illustrated which can be used for creating a dashboard from the associated model. The idea behind this framework is to make the decision support system highly data-driven. This also makes the system easily accessible and usable for the problem owner, without changing much in the underlying optimization model/data.

Summary

In this chapter, the business problem is formulated in AMPL which is a well-established AML. Model instantiation and solution from the solver are obtained and demonstrated. Two approaches for results analysis are discussed. The data-driven analysis gives rise to the idea of having a master coefficient for the optimization model. This will be used as a reference for the proposed framework. A subset of this data can be used to represent coefficients for a different business scenario. The role-based analysis explains the interaction of different user roles with components of DSS. This gives rise to the idea of a system which supports multi-user access. Based on the role, users can access and use the same system for the purpose of analysis. A detailed example of this is explained in chapter 4.

4 An Interactive DSS Framework

A real-world optimization system can include data, inferencing model and a solver as its components. However, an integrated decision support system comes with the ability to derive new knowledge from optimization components. Optimization components can be considered as knowledge systems in a generic DSS as proposed by Holsapple et al. 1988. Typical systems can provide the output as a response to the input request by the decision maker and can act as an aid in making decisions.

Until this point, the development of the optimization model, data and its solution are explained. In this chapter, an overview of a framework is proposed which makes effective use of the model and data. The framework uses the findings from the previous chapters. It uses the master coefficients data-set that was discussed earlier. Also, the idea behind the framework is to create a system which can benefit different users in an optimization-based DSS. A web-platform is chosen since it is a modern-day standard and it makes remote access possible.

4.1 Overview of the System Architecture

Figure 4.1 illustrates the system architecture diagram for an interactive DSS framework with an optimization model. The main component of the framework proposed in this case-study is an API by AMPL (*AMPL API* n.d.), which is used to communicate with the modeling system. The decision maker usually has access to all input datasets. Additionally, the decision models are created in an intelligent way such that certain configuration parameters are data-driven. By changing meta-data and configuration data different realizations of model instances can be created as required. The underlying

framework reads the model and automatically creates an interactive dashboard. The dashboard comes with a rich set of input controls and can be used for changing the input data as well as configuration. Since the decision model is data-driven and data components are separated from the model components, all configurations of the model can be controlled by the input dashboard controls. This eliminates the necessity of changing the underlying data file for each input configuration.

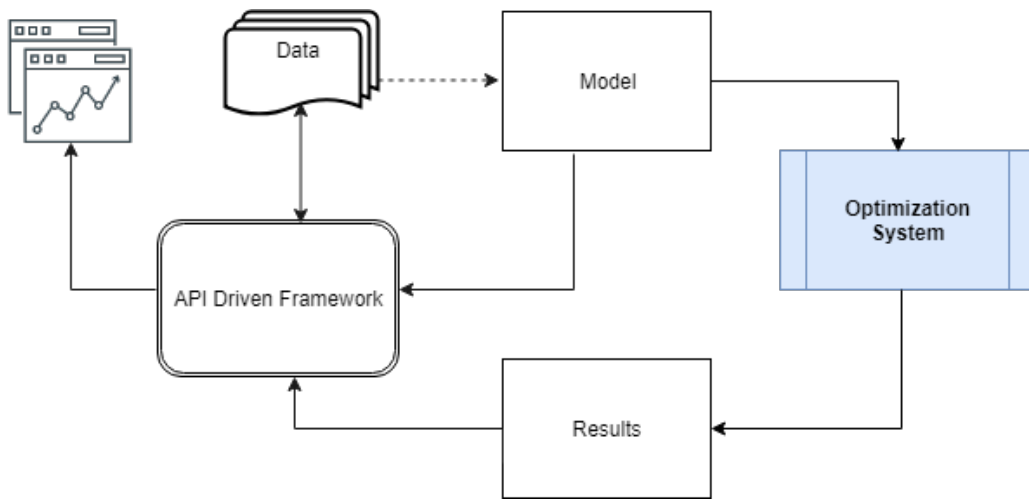


Figure 4.1: System Architecture

For each set of configurations, the instantiated model interacts with the solver system and yields a solution to the problem. Once the model is solved the output results are required for analysis by the decision maker. So it is essential to store these results. As a standard practice, we have transformed the results data of the mathematical programming model to a relatively easy-to-analyse multidimensional database. We believe the results and input data are closely related and can be best represented in terms of a relational database. Furthermore, we have converted all the associated datasets into multidimensional tables so that the OLAP capability can be fully exploited.

The dashboard also comes with the ability to convert these data views to data visualization components like graphs, charts etc. These are an essential aid to corporate management and the decision-making process.

4.2 An Interactive Dashboard for the Model

As discussed in section 4.1, the decision model presented in this research is part of the proposed framework. The framework automatically reads the mathematical model (AMPL) and creates an interactive dashboard. The model needs to be instantiated before it gets loaded into the dashboard by the API. API in python is used in this case.

The task of the API is broken down and given below:

- (i) Parsing the model instance: In this step, the model file and data file are parsed by the API. The API reads the location of these files and examines the model instance. If there is any error it gets propagated to the dashboard. Otherwise, the model and the data file get loaded in AMPL and a model instance is ready for the next set of actions. An example of the supply chain model is given in a later section.
- (ii) Components of the model : Various components of the model are collected in this step. Similar components get stored in a list. Different lists store different types of model components and their values such as sets, parameters, variables, objective and constraints. The API also reads the way these entities are declared. This helps in making decisions about the interactive components.
- (iii) Solver interaction : The response from the solver is collected by the API from the underlying AMPL process. The response includes detailed

component lists and its values from the model instance. We have implemented various methods to read this response and create the expected input dashboard components.

The API is also responsible for converting the results of the decision model into multidimensional data tables(views). These data views are integrated with the output dashboard for further analysis. The Multidimensional Database (MDDDB) structure also helps in the representation of data in terms of visualization components like graphs and charts etc.

4.3 Instantiation with Master Dataset

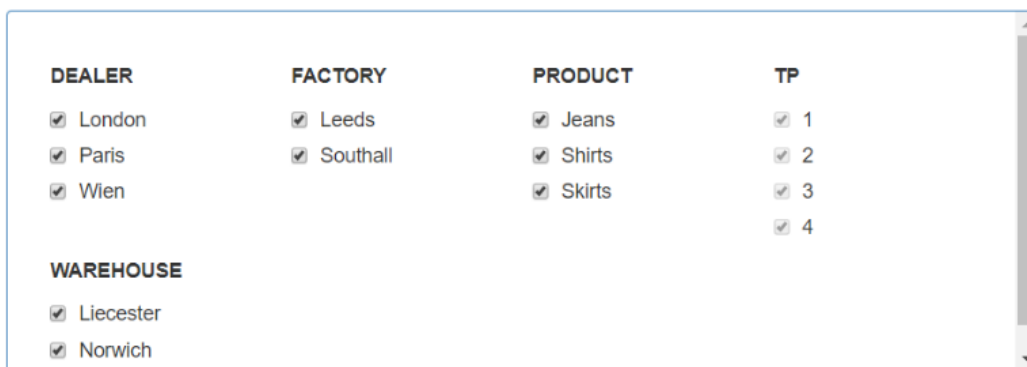
It has been previously established that the decision model presented here is separate from the data. The separate model file and data files in AMPL are presented in Appendix B.

In section 3.2.1, we have introduced a concept of a master set of coefficients which can be used for a decision model. The decision maker has access to all input datasets during the investigation of the problem. Thus, all the possible input values are included in the data file. These datasets can be considered as a global or master list with all the feasible values. The proposed framework is efficient enough to produce modified and/or a subset of the original data-points. This helps in better dashboard management and flexible configuration option selection. To avoid reloading of the entire input dashboard, a default instantiation of the model is done using the master dataset.

4.4 Dashboard Controls of Model Components

We introduce an example of dashboard control for the supply chain model as discussed in the earlier section. Model components from the input dashboard are shown in a web browser interface. This ‘thin-client’ dashboard uses modern UI components created by the help of standard JavaScript libraries.

- **Sets:** From the AMPL model file as given in Appendix B, we have identified five different sets. Furthermore, values for these sets are given in Table 3.1. The input dashboard shows these values as given in Figure 4.2. Since the dashboard is interactive, the problem owner has the freedom to choose any data member of these sets. Therefore each member of the set is represented with a check-box style control. It can be unselected in-order to discard the member of the set.



The screenshot shows a web interface with four columns of model sets, each with a title and a list of members with checkboxes. The sets are DEALER, FACTORY, PRODUCT, and TP. There is also a WAREHOUSE section below the others.

DEALER	FACTORY	PRODUCT	TP
<input checked="" type="checkbox"/> London	<input checked="" type="checkbox"/> Leeds	<input checked="" type="checkbox"/> Jeans	<input checked="" type="checkbox"/> 1
<input checked="" type="checkbox"/> Paris	<input checked="" type="checkbox"/> Southall	<input checked="" type="checkbox"/> Shirts	<input checked="" type="checkbox"/> 2
<input checked="" type="checkbox"/> Wien		<input checked="" type="checkbox"/> Skirts	<input checked="" type="checkbox"/> 3
			<input checked="" type="checkbox"/> 4

WAREHOUSE

- Leicester
- Norwich

Figure 4.2: Model Sets

- **Parameters (Scalar):** We classify the parameters of the model into two categories. Scalar and non-scalar (indexed) parameters. The scalar parameters of the model are also part of the input dashboard as shown in Figure 4.3. Some of the parameters ($maxcost_{fd}$, $maxcost_{fw}$,

maxcost_wd) are derived from a given dataset. Hence the values shown in the figure below is read-only. It cannot be updated from the dashboard. However, the dataset can be modified from the data source to change the value of derived parameters. The values of certain parameters are dependent on other entities of the model, i.e changing the value of the parameter requires addition or deletion of indices to other sets/parameters. One example is *NT* which is the total number of time intervals in the model. Hence, it determines the cardinality of the set that represents the time periods. Such parameters are also uneditable in the dashboard. During the parsing of a model instance, the API makes the decision on interactive and read-only parameters.

NT	4
maxcost_fd	2.6
maxcost_fw	0.3
maxcost_wd	1.3

param dealerdemand{p in PRODUCT, DEALER, TP} >= 0;

index0	index1	index2	dealerdemand
Jeans	London	1	20
Jeans	London	2	55
Jeans	London	3	66
Jeans	London	4	76

Figure 4.3: Parameters

Figure 4.4: Indexed Parameter

- Parameters (Indexed):** These parameters of the model are presented in a tabular format. Each indexed parameter table consists of two types of columns. One set of columns represent the indexing set over which the parameter is dimensioned. The value of the parameter is represented in a separate column. We have used a different colour code in order to make this identification of the type of column. Interactive columns of the table are represented in green whereas non-editable (indexing) columns are made grey. Indexed parameters of the dashboard are shown

in Figure 4.4. Parameter *dealerdemand* has three indexing sets. The header of the table shows the declaration statement which can be related to table headers in the figure below [*index0, index1..*].

- **Model Objective** : The linear objective function of the model is also part of the dashboard. The objective function as declared in the AMPL model can be identified in the complete dashboard in Figure 4.7. The model objective and the function type (maximize or minimize) is included.
- **Solver Output** : The instantiated model from the dashboard can interact with the underlying solver. The solver output(messages) are read using the API and are available in the solver panel of the dashboard. Figure 4.5 shows the solver panel. The boolean status on top shows the success condition. A detailed message is available in the bottom panel. In the case of an error or failure, an appropriate message is shown in the solver panel. An example error status and a message is shown in Figure 4.6.

```
Success : [true]

Solver Message : CBC 2.8.8: ██████████CBC 2.8.8 optimal, objective 10181.26667 129 iterations
```

Figure 4.5: Solver Panel Success

```
Success : [false]

Solver Message : line -1 offset -1 Error executing "let" command: can't convert '3wewe' to a number.
```

Figure 4.6: Solver Panel Error

4.4.1 Load Model Configuration

Different dashboard components for the model are described in section 4.4. In order to make the interactive DSS work, a certain configuration is required for the framework. The optimization model and data components are represented in separate files. The framework needs to access the location of these files. For this purpose, a configuration file is exposed to the problem owner. This file acts as a single point of entry for all the system configuration. A sample configuration file is shown in Figure A.1 under section A.2 in Appendix A.

Once the configuration file is populated with valid entries, the model can be instantiated and loaded fairly easily from the dashboard. In the previous section, we discussed the idea of default instantiation of the model using the master data set. This is achieved by configuring the data component options of the configuration file. A button panel is present in the dashboard which consists of two buttons as shown in Figure 4.7. Clicking on '*Load Model*' instantiates the model and populates the dashboard with the master dataset.

4.4.2 Run and Solve Different Instances

A set of algorithms and procedures are followed by different solvers for solving an optimization model. AMPL supports a list of solvers and these can be easily hooked into any AMPL system. A typical solving process in AMPL involves loading of the model file and the data file first. Then the *solve* command starts a series of activities for solving the model. The AMPL session needs to be reset every time before reinitialization of the model. A series of commands for solving a model are given below. Any conditional statement, reassignment and other control statements may appear in between if required.


```
ampl: model modelFile.mod;
ampl: data dataFile.dat;
ampl: solve;
ampl: CPLEX 12.6.0.0: optimal solution;
objective 10270.2146
128 dual simplex iterations (0 in phase I);
```

By use of the API, we have introduced a minimal form of solver interaction from the dashboard. This brings drastic improvement to the time taken for the total solving process. The rich dashboard controls provide single one-click model instantiation, which helps in better understanding and investigation of the problem under analysis. Clicking on *'Load Model'* loads the model components from the configuration file. Similarly clicking on *'Solve Model'* sends the loaded model instance to the underlying solver for processing. The output from the solver is available under the solver panel as shown in Figures 4.5 and 4.6. The buttons for loading and solving the model can be identified in the dashboard Figure 4.7.

4.4.3 A Consolidated View

Figure 4.7 represents the complete view of the dashboard. The top part of it includes the input locations loaded by the configuration file. The sets are included after that followed by the parameters section. The parameter section includes the scalar parameters as well as indexed parameters in the form of interactive data tables. The section below includes the objective function of the underlying model. Two buttons are placed below, for loading and solving of the model instance. A solver panel is present to display the message from the solver. A button *'Explore AMPL Entities'* is also present which takes the problem owner to the Business Intelligence platform. Further information

about this platform is explained in chapter 6.

Summary

This chapter presents a framework which explains the creation of a web-based dashboard for an optimization model. It may act as a useful tool for building a web-based dashboard platform for an optimization problem. This platform represents the components of the optimization system in terms of modern UI widgets. Furthermore, making the system interactive ensures any users can operate it without any expertise in optimization components. Typical operation at this point can include, instantiation of the system, creating different instances of a model, and solving the instances. This type of set up is still missing the analytical aspect of a DSS. Hence the next set of tasks include adding a supporting analytical platform. The analytic platform can make use of optimization data and results.

Supply Chain Management

AMPL DIR - MODEL - sp1c2.mod DATA - sp1c2.dat
 C:\Users\Ansuman\Desktop\Amp\l\amp1dev\SPLC_V2\model\

DEALER	FACTORY	PRODUCT	TP
<input checked="" type="checkbox"/> London	<input checked="" type="checkbox"/> Leeds	<input checked="" type="checkbox"/> Jeans	<input checked="" type="checkbox"/> 1
<input checked="" type="checkbox"/> Paris	<input checked="" type="checkbox"/> Southall	<input checked="" type="checkbox"/> Shirts	<input checked="" type="checkbox"/> 2
<input checked="" type="checkbox"/> Wien		<input checked="" type="checkbox"/> Skirts	<input checked="" type="checkbox"/> 3
			<input checked="" type="checkbox"/> 4

WAREHOUSE

Leicester

Norwich

NT param dealerdemand{p in PRODUCT, DEALER, TP} >= 0;

	index0	index1	index2	dealerdemand
4	Jeans	London	1	20
maxcost_fd 2.6	Jeans	London	2	55
maxcost_fw 0.3	Jeans	London	3	66
maxcost_wd 1.3	Jeans	London	4	76

<p>Decision Variables</p>	<p>Objective</p> <p>minimize cost_To_Mnfc: costTot;</p>
---------------------------	--

Load Model
Solve Model

Success : [true] Explore AMPL Entities

Solver Message : MINOS 5.51: optimal solution found. 152 iterations, objective 10181.26667

[Cubes Server Info](#)

[AMPL Entities List \(Cubes\)](#)

Figure 4.7: Supply Chain Dashboard

5 Model Data Analysis

Section 3.1 explains the formulation of the linear programming problem in AMPL. The input and results of such problem consist of a number of sets and indices which resemble closely to any relational database system (Mitra et al. 1995). The multidimensional structure of associated data can be used for analysis, data mining, problem-solving and knowledge discovery. In short, OLAP approach (Edgar F Codd et al. 1993) helps in exploiting this multidimensional analytical database and making decisions over a long time horizon.

This chapter covers a brief introduction to the relational database management system and the evolution of the multidimensional database. Efficient decision models use an analytical multidimensional database as a source of input data. The structural resemblance with data-cubes and its relation with model data is explained in brief. Various OLAP features are demonstrated with model data which helps in ‘*what-if*’ analysis (Philippakis 1988).

5.1 Evolution of Relational DBMS

Database management as such did not go through a formal evolution process until the relational model introduced by E. F. Codd 1970. Prior to this, database management was application specific and often written in COBOL. As per the relational model, data is presented in a number of tables (relations). Each table comprises of tuples and attributes. A tuple is equivalent to a table row consisting of an ordered list of elements. The tuple has its own key which is unique for identifying a record. In a Database Management System (DBMS), this is called a unique primary key. Multiple relational tables use a primary-key and a foreign-key mechanism for cross-referencing. A

foreign-key in a table is nothing but a value that matches with a primary-key of another table. Attribute refers to a table column which represents a set of data values of a specific type. Figure 5.1 represents the structure of a table.

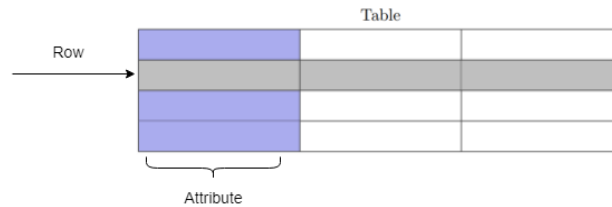


Figure 5.1: Structure of Relational Table

The relational data model resulted in the development of various database management systems over the next few years (Edgar F Codd 1979, Edgar F Codd 1990). The concept of relational algebra provided a theoretical underpinning for query languages. Codd described one such language known as Structured Query Language (SQL) for database programming. SQL commands enable database users to access and modify records without specifying retrieval details. Over the next few decades, SQL evolved to support advanced data types and complex analysis. It became the commercial standard for relational database management system (RDBMS). Detailed concepts of the relational database system are given in the book by Date 1990.

In recent times internet and web-based technology had a revolutionary impact on database management systems. DBMS are supported as an integrated application suite with a robust client-server architecture. Service-oriented computing has made database services available as cloud services. Some of the DBMS software can also be embedded in a stand-alone client application.

5.2 Multidimensional Databases

As the corporate data increased exponentially over time, analysis of the data became essential. The evolution of multidimensional database structure made the data analysis relatively easy for information specialists. Data analysis became a critical factor for extracting useful business information. In the past, the primary source of information storage was a transactional database. These are called OLTP (On-line Transactional Processing) systems. However, these systems lack measurement and aggregation of data in a particular application. This led to the requirement of a data analysis process called multidimensional analysis. The target database for this purpose is known as an analytic database. In the late 1980s, the enterprise data warehouse (EDW) came into the picture which consists of a consolidated database from diverse sources and is ideal for specialized analysis. The EDW ensures storage of gigantic sets of data from multiple transactional databases. Analysis of this database plays a key role in the decision making process. The set of tools which provide answers to business queries from large aggregated databases are known as OLAP tools. OLAP analysis includes multidimensional tables in the form of data cubes. The cube hierarchy supports operations like aggregation, roll up, drill down across multiple dimensions. We explore these features of OLAP and demonstrate how they can be integrated swiftly to an underlying DSS.

A multidimensional table of data can be represented as a cube in an OLAP system. A *fact* in an OLAP system is a value of the most detailed unit of data. The OLAP data cubes can have any number of *dimensions*. Aggregation, filtering, sorting of data is possible across each dimension of a cube. A detailed analysis of fact and dimensions in a data warehouse is given in Rowen et al. 2001 and Kimball et al. 2011.

We have used an open source software framework called *Cubes* (*Cubes* n.d.) in this case study. It is a well-established OLAP tool written in python (*Python Programming Language* n.d.) for reporting in analytical applications. *Cubes* support multidimensional expressions (MDX queries) and aggregated browsing of multidimensional data which is ideal for a DSS. The dataset used in the DSS can be classified into two groups. i.e. input data and output(data) results. The dataset which is known to the problem owner at the start is part of the input data. This input data is used in the AMPL model and corresponding data files are given in Appendix B. The output results are obtained from the solver for a particular model instance. These values get assigned to the decision variables of the AMPL model. We have created a centralized database which stores the input data as well as output results. Figure 5.2 shows a centralized SQLite database, which stores input as well as output data. Various tables in this database can be considered and presented as OLAP cubes.

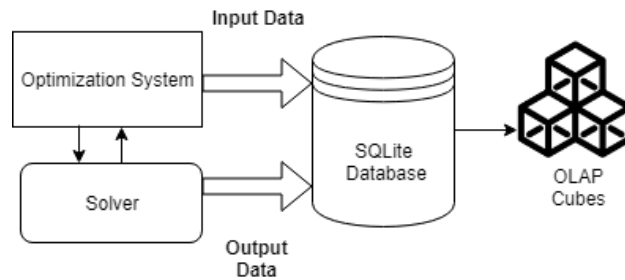


Figure 5.2: OLAP Cube from Data

An OLAP Cube

Let us consider an input parameter of the model, dealer demand. The corresponding values for AMPL parameter *dealerdemand* are set out in Table

A.7 in Appendix A. The parameter is indexed over three sets. From the model components declaration, *dealerdemand* is given as, DE_{pdt} : Expected Demand for product p at dealer d at time t . A typical OLAP cube for dealer demand is given in Figure 5.3 below. Considering the values as measure and three indexing sets as the dimensions of the cube, we will evaluate various ‘what-if’ conditions across these dimensions. Each dimension can be further subdivided into multiple hierarchies. e.g. the time dimension can be daily/monthly/yearly/quarterly. Various OLAP features described here are helpful in evaluating different business cases. These features are later used in a data visualization system for better analytical and reporting purposes.

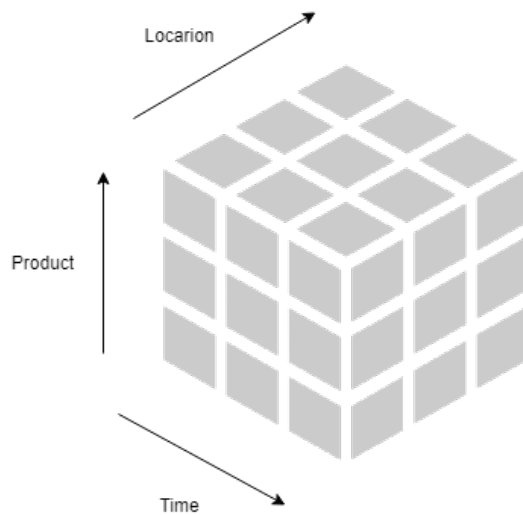


Figure 5.3: OLAP Cube *Dealerdemand*

5.3 OLAP Features Illustrated with Model Data

The term OLAP *Cube* refers to a multidimensional dataset. Although a cube has three sides, practical cases of *cubes* can have any number of dimensions. Sometimes it is also referred to as a *hypercube* when the number

of dimensions exceeds three. To facilitate analysis, various conceptual features of the OLAP *cube* can be exploited. A few of these features are described below. Any thin-client based system (browser) can make requests to the ‘*Cubes*’ framework in the form of a Uniform Resource Locator (URL). The response from ‘*Cubes*’ is in the form of JavaScript Object Notation (JSON). JSON is simple text in an open-standard file format, easily readable and consists of key-value pairs. More details of JSON data type is given in Crockford 2009.

5.3.1 Aggregated Browsing

We start by exploring the aggregation(roll-up) feature of OLAP. Both aggregated and disaggregated data can be obtained from the *dealerdemand* table. Aggregated results give better insight when summary information is required. In this example, the sum of the total number of demand is the aggregated value. This information may be useful to senior management. In larger models, it is often useful to know the aggregated data for better resource utilization.

The aggregation task is taken care of by the *Cubes* framework. The framework proposed in this research automatically generates a *data cube* from the model parameter under consideration. One can simply query the OLAP system to get the aggregated result. The analytical functionality of the framework is provided by requests. A sample request for this case is given below.

```
Request:/cube/dealerdemand/aggregate
```

```
{  
  "summary": {  
    "Aggregate_dealerdemand": 1350.0
```

```

    },
    "remainder": {},
    "cells": [],
    "aggregates": [
        "Aggregate_dealerdemand"
    ],
    "cell": [],
    "attributes": [],
    "has_split": false
}

```

Listing 1: Aggregated Result from Cubes

5.3.2 Drill Down Data

Drill-down across the dimension is required when the problem owner is interested in detailed information. While it may be useful for senior management to look at the total aggregated number, factory managers will be more concerned with the drilled down data. Drilling down across a location might be an important piece of information to consider for a regional manager. Both the aggregated and disaggregated data are nothing but different information extracted from the same data source by OLAP tools. The generated data cube can provide the problem owner with a disaggregated dataset based on one or more dimensions. In our example, there are three dimensions available for the *dealerdemand* cube.

Drill down across a dimension : Table 5.1 represents the aggregated data drilled down by dimension product. A sample request to the *Cubes* framework and response is shown in listing 2. Only the relevant portion of the response

is represented below. The analytical functionality of the framework provides more information like levels, hierarchy, attributes etc.

Request: `/cube/dealerdemand/aggregate?drilldown=dealerdemand_p_in_PRODUCT`

```
{
  "summary": {
    "Aggregate_dealerdemand": 1350.0
  },
  "remainder": {},
  "cells": [
    {
      "dealerdemand_p_in_PRODUCT.index0": "Jeans",
      "Aggregate_dealerdemand": 655.0
    },
    {
      "dealerdemand_p_in_PRODUCT.index0": "Shirts",
      "Aggregate_dealerdemand": 443.0
    },
    {
      "dealerdemand_p_in_PRODUCT.index0": "Skirts",
      "Aggregate_dealerdemand": 252.0
    }
  ],
  "total_cell_count": 3
}
```

Listing 2: Drilled down Result from Cubes

A typical pivot table with two drilled down dimensions is also shown in

Product	Sum of Demand
Jeans	655
Shirts	443
Skirts	252

Table 5.1: Drill Down across One Dimension [Product]

Table 5.2. The grand total showing aggregated data through columns and rows.

Sum of Demand				
	Region			
Product	London	Paris	Wien	Grand Total
Jeans	217	296	142	655
Shirts	247	49	147	443
Skirts	78	78	96	252
Grand Total	542	423	385	1350

Table 5.2: Drill Down across Two Dimensions

Drill down across all dimensions : Figure 5.4 represents the disaggregated data cube with all dimensions. Each cell of the cube contains a value. The request to the cube framework to get the drilled down data and the response is shown in listing 3 below. It is very difficult to represent the cube when the dimensions exceed three. However, the OLAP framework can handle hundreds of dimensions with each dimension having multiple sub-levels. The response JSON is easy to interpret and can be used by any reporting and

analytical tools.

```
Request:/cube/dealerdemand/aggregate?drilldown=
dealerdemand_p_in_PRODUCT&drilldown=dealerdemand_
DEALER&drilldown=dealerdemand_TP
```

```
{
  "summary": {
    "Aggregate_dealerdemand": 1350.0
  },
  "remainder": {},
  "cells": [
    {
      "dealerdemand_p_in_PRODUCT.index0": "Jeans",
      "dealerdemand_DEALER.index1": "London",
      "dealerdemand_TP.index2": 1.0,
      "Aggregate_dealerdemand": 20.0
    },
    {
      "dealerdemand_p_in_PRODUCT.index0": "Jeans",
      "dealerdemand_DEALER.index1": "London",
      "dealerdemand_TP.index2": 2.0,
      "Aggregate_dealerdemand": 55.0
    }
  ],
  "total_cell_count": 36,
  "aggregates": [
    "Aggregate_dealerdemand"
  ]
}
```

Listing 3: Disaggregated Result from Cubes

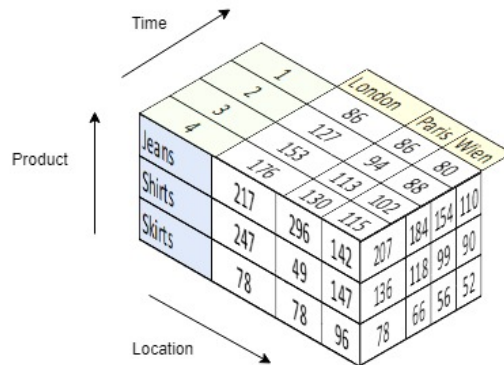


Figure 5.4: Aggregated OLAP Cube

5.3.3 Slicing and Dicing

Slicing means fixing a single value across one dimension of a cube resulting in a rectangular cut to represent the subset of the cube. In our example, let us assume the data for dealerdemand is required for a single region (London) only. Then the ‘sliced’ out portion of the cube is considered. Furthermore, the problem owner can choose whether aggregated or drilled down data is required. A sample request and response for the slice is shown below in listing 4. The location dimension is fixed in this case and thus only shows data for ‘London’. The results are drilled down data across products and time period. Figure 5.5 represents the sliced data cube.

Request:/cube/dealerdemand/aggregate?drilldown=dealerdemand_p_in_PRODUCT&drilldown=dealerdemand_TP&cut=dealerdemand_DEALER:London

```
{
  "summary": {
    "Aggregate_dealerdemand": 542.0
  },
}
```

```

"remainder": {},
"cells": [
  {
    "dealerdemand_p_in_PRODUCT.index0": "Jeans",
    "dealerdemand_TP.index2": 1.0,
    "Aggregate_dealerdemand": 20.0
  },
  {
    "dealerdemand_p_in_PRODUCT.index0": "Jeans",
    "dealerdemand_TP.index2": 2.0,
    "Aggregate_dealerdemand": 55.0
  },
  {
    "dealerdemand_p_in_PRODUCT.index0": "Jeans",
    "dealerdemand_TP.index2": 3.0,
    "Aggregate_dealerdemand": 66.0
  },
  {
    "dealerdemand_p_in_PRODUCT.index0": "Jeans",
    "dealerdemand_TP.index2": 4.0,
    "Aggregate_dealerdemand": 76.0
  },
  {
    "dealerdemand_p_in_PRODUCT.index0": "Shirts",
    "dealerdemand_TP.index2": 1.0,
    "Aggregate_dealerdemand": 50.0
  }
]
}

```

Listing 4: Sliced Result from Cubes

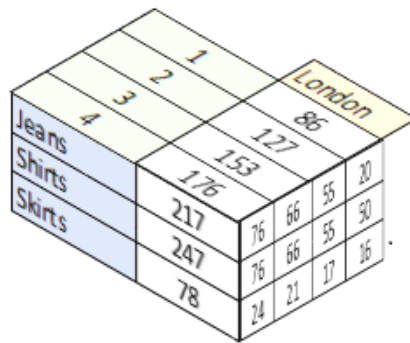


Figure 5.5: Cube *Dealerdemand* with Sliced Data

Dicing means picking a filter across a single or multiple dimensions of the cube. The new sub-cube shows filtered data. At times the problem owner might be interested in a portion of the cube data along a dimension. In this instance, let us assume the drilled down data is required for only the first two time periods with the remaining dimensions intact. Hence we added only this range (time period 1 & 2) to the query. The drill down parameters are the same. The resulting request query and response to the cube framework is shown in listing 5. Figure 5.6 presents the diced cube for the first two time periods.

```
Request:cube/dealerdemand/aggregate?drilldown=
dealerdemand_p_in_PRODUCT&drilldown=dealerdemand_TP&
drilldown=dealerdemand_DEALER&cut=dealerdemand_TP:1;2
```

```
{
  "summary": {
    "Aggregate_dealerdemand": 561.0
```



```

},
"remainder": {},
"cells": [
  {
    "dealerdemand_p_in_PRODUCT.index0": "Jeans",
    "dealerdemand_TP.index2": 1.0,
    "dealerdemand_DEALER.index1": "London",
    "Aggregate_dealerdemand": 20.0
  },
  {
    "dealerdemand_p_in_PRODUCT.index0": "Jeans",
    "dealerdemand_TP.index2": 1.0,
    "dealerdemand_DEALER.index1": "Paris",
    "Aggregate_dealerdemand": 60.0
  },
  {
    "dealerdemand_p_in_PRODUCT.index0": "Jeans",
    "dealerdemand_TP.index2": 1.0,
    "dealerdemand_DEALER.index1": "Wien",
    "Aggregate_dealerdemand": 30.0
  },
  {
    "dealerdemand_p_in_PRODUCT.index0": "Jeans",
    "dealerdemand_TP.index2": 2.0,
    "dealerdemand_DEALER.index1": "London",
    "Aggregate_dealerdemand": 55.0
  }
]
}

```

Listing 5: Diced Result from Cubes

	1	London	Paris	Wien
Jeans	2	86	86	80
Shirts		127	94	88
Skirts		75	126	63
		105	21	63
		33	33	42
		154	99	90
		110	56	52

Figure 5.6: Cube *Dealerdemand* with Diced Data

Summary

The results from an optimization model may be in a format more suitable for the optimization system and its components. However these results can be transformed into multidimensional data-cubes for effective analysis. This chapter provides an approach which shows the OLAP feature in use to analyse output (results) and answer different business questions in the DSS. Using such features require certain level of expertise with the database management system. Hence, an integrated BI platform is proposed in the following chapter. The web-based dashboard can easily integrate with the BI platform. Furthermore, the use of the BI platform is similar to the web-based dashboard and does not require expertise in database analysis.

6 Data Visualization and Business Intelligence

Visualization is a key aspect of human communication. All business intelligence products are derived from this assumption and therefore have embedded ‘visualization’ functionality in one way or another. In the concept of experimental data analysis, there are namely two aspects of visualization. *(i)* Multidimensional data views or structural views of data and *(ii)* Chart and graph illustration of data. Many modern-day BI platform includes these data visualization facilities.

In this chapter, we explain how data visualization is tightly coupled with DSS and OLAP tools. The OLAP framework discussed in the previous section is highly efficient in handling multidimensional data-cubes. However, queries to an OLAP system requires an in-depth understanding of database systems. To overcome this, a BI platform with embedded visualization system is discussed in detail. OLAP features are demonstrated from the visualization platform. A practical business problem and possible solution approaches using the framework are also discussed.

6.1 A Visualization System for OLAP

Any OLAP framework is generally capable of handling all Multidimensional Expressions(MDX) queries, however it is difficult to analyse the structure of the relational database when a large amount of data is present. Hence a data visualization tool is an aid to decision makers and analysts. Maniatis et al. 2003 presented some advanced visualization techniques for OLAP. An interactive technique for analysis of multidimensional databases is given by Techapichetvanich et al. 2005. Visualization of OLAP features like drill down from aggregation is presented in Mansmann et al. 2007. In general, a BI

platform with embedded visualization system gives more context to the data and results in a DSS as compared to visualization system alone. A generic approach to build a Business Intelligence system is discussed in Olszak et al. 2007. Yeoh et al. 2010 research shows some critical success factors for a BI system.

This research sets out an embedded visualization platform for OLAP which can be useful for data analysis without needing a certain level of expertise on the modeling tool and the OLAP tool. The proposed framework integrates the data visualization system with the web-based dashboard. It automatically instantiates the OLAP data-cubes for the underlying decision model. A single web-based dashboard in this way can manage the input to the DSS as well as the output. The data visualization system makes the request to the OLAP framework and shows the response in the required format as chosen by the problem owner.

We have used the *Cubesviewer* (*CubesViewer* n.d.) framework in this case-study. It is an open source library for data exploration and reporting. It also supports multidimensional datasets and embedding of visual components like graphs and charts. In conjunction with the OLAP tool, *Cubesviewer* can serve as an analytical tool for end users as well as business owners. The platform for a visualization system can be accessed from the dashboard as explained in chapter 4. Any cube can be selected by the problem owner for further analysis from the list of cubes as given in Figure 6.1.

AMPL Visualization

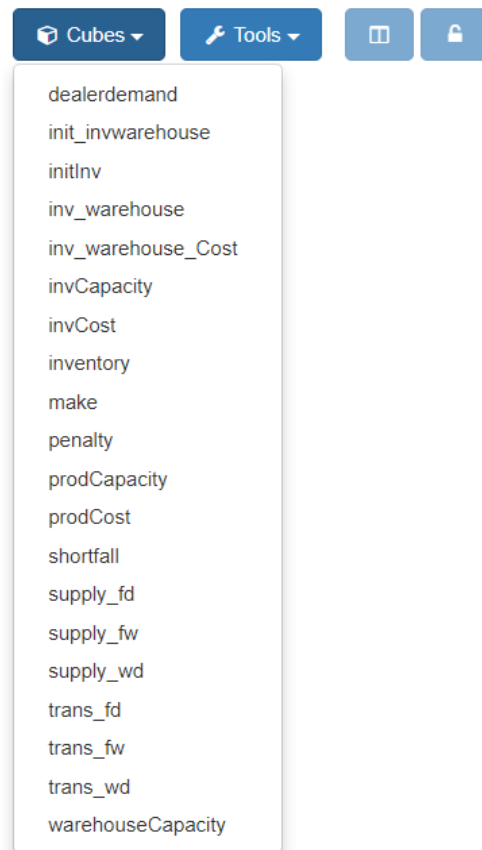


Figure 6.1: Available Cubes

6.2 Data Views (Tables)

A natural way to represent the structure of data is by means of data tables. Although at times the analyst may seek a certain form of data visualization component instead of tables. Data tables show the information by means of rows and columns. The structural resemblance of a data table can be compared to a relational database in most cases. In this research, all the data visualization components are closely coupled with the OLAP framework.

Thus there are mainly two types of data tables under consideration. i.e aggregated tables and detailed tables.

- *Aggregated* data tables show the aggregated form of multidimensional data. Let us consider a parameter in the model which corresponds to the initial inventory values for different products. In Table 2.2 the parameter is given as IF_{pf} . $initInv$ denotes the corresponding parameter in the optimization model. Figure 6.2 shows the aggregated data table for initial inventory data. By default, any cube selected from the list as given in Figure 6.1 produces the aggregated data table view.

Init Inv	Sum Of Init Inv
Summary	25.00
25.00	

Figure 6.2: Aggregated Data Table for $initInv$

- *Detailed* data tables show the summary across all dimensions. The list of dimensions are available under the drill down option and can be selected as shown in Figure 6.3. If we consider the same parameter IF_{pf} , the detailed summary table is given in Figure 6.4. The drilled down dimensions are product and factory which are shown as two columns.

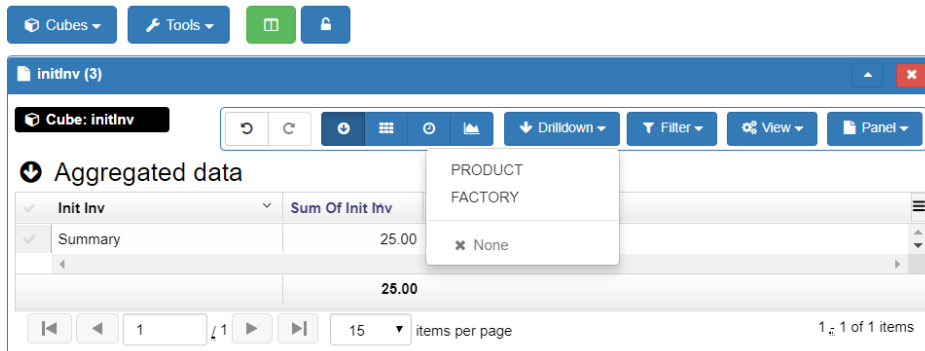


Figure 6.3: Dimension for Data Table *initInv*

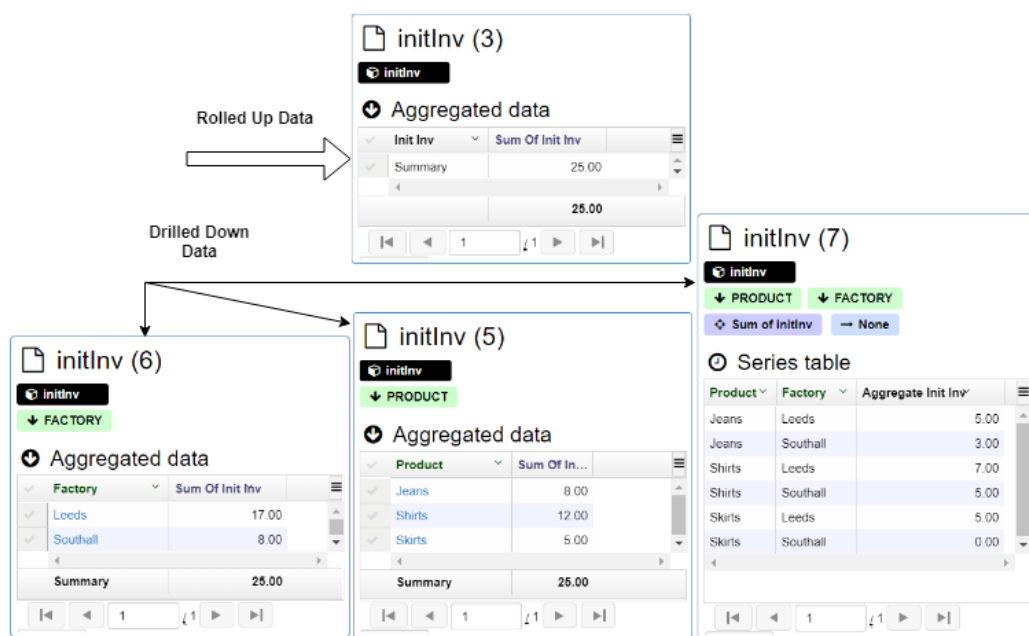


Figure 6.4: Detailed Data Table *initInv*

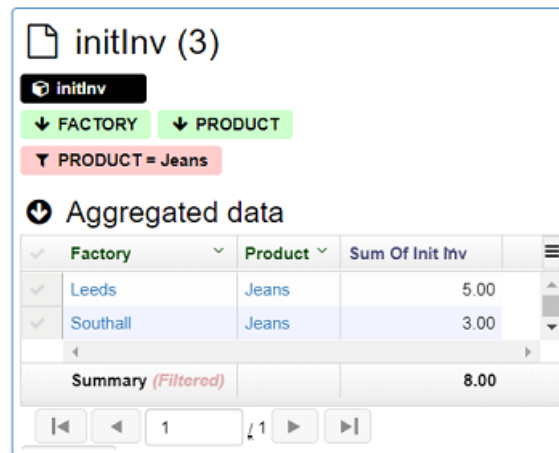
6.2.1 Drill Down

The data visualization library also provides a rich set of controls which are very easy to use. Once a cube is selected for analysis, the aggregated data is available as given in Figure 6.2. The detailed table is given in Figure 6.4 can be generated from the aggregated table shown in Figure 6.2. The list

of dimensions available to the analyst is in Figure 6.3. The analyst has to choose the drilled down dimensions to get detailed data for it. The available dimensions for drill down are *Product* and *Factory*.

6.2.2 Filter Data

There is a set of controls which can be used to filter data further as shown in Figure 6.3. The concept of slicing and dicing was explained in the previous section. The visualization library provides easy to use control options for this. Aggregated data as well as drilled down data can be filtered further. The data table in Figure 6.5 shows the filtered drilled down data. The product under consideration is *jeans* only. Thus it is chosen as the filter option. A summary of the filtered data is also present in the data table.



Factory	Product	Sum Of Init Inv
Leeds	Jeans	5.00
Southall	Jeans	3.00
Summary (Filtered)		8.00

Figure 6.5: Filtered Data Table

6.3 Graphs and Charts

Charts and Graphs are a natural way of illustrating digital information in a diagrammatical form. For reporting and analysis purposes an analyst

may require different types of graphs and charts. The format of the results presented to senior management may differ from those presented to the production manager. The visualization components for both cases are different. Various types of charts and graphs supported by the framework are (i) *Pie*, (ii) *Bar*, (iii) *Line*, (iv) *Area* and (v) *Radar*. These options are available under the ‘view’ menu as given in Figure 6.6. Two essential options that need to be selected for charts are the horizontal dimension and the measure entity.

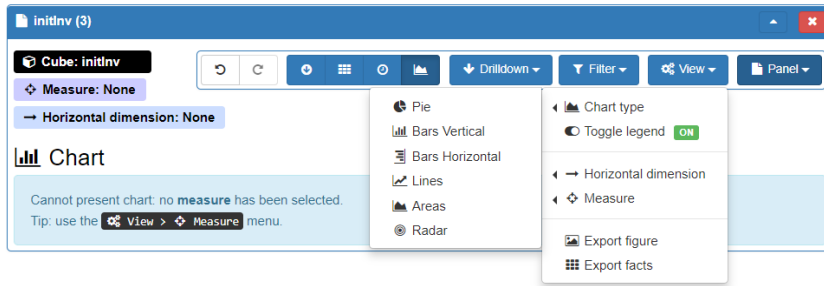


Figure 6.6: Chart Options

For illustrative purposes, some examples are shown here using bar charts and line charts. However, this applies to any other visualization component. Some of the control options are also hidden for simplicity purposes. Let us consider parameter q_{pft} from Table 2.2 which represents the amount of product to be shipped from a factory to a warehouse in a certain time period. For any visualization component, there are mainly two selections required. The measuring parameter and the dimension (horizontal). Figure 6.7 represents the q_{pft} , where the measuring parameter is along the vertical axis and the horizontal parameter is the product. Further drill down of the data is also possible across the remaining dimensions. The set of controls for the selection of these is given in Figure 6.6.



Figure 6.7: Supply from Factory to Warehouse [Aggregated]

6.3.1 Drill Down

As shown in the data table, the drill down feature can be exploited in other data visualization components like graphs and charts. A simple bar chart of parameter q_{pfwt} is shown in Figure 6.7. The horizontal dimension shows the drilled down data for products. Further drill down across dimensions like warehouse will produce a graph as given in Figure 6.8

6.3.2 Filter Data

Similar to the data table, the filtering of data in a chart can be performed easily. The drilled down data across factory and product is given in Figure

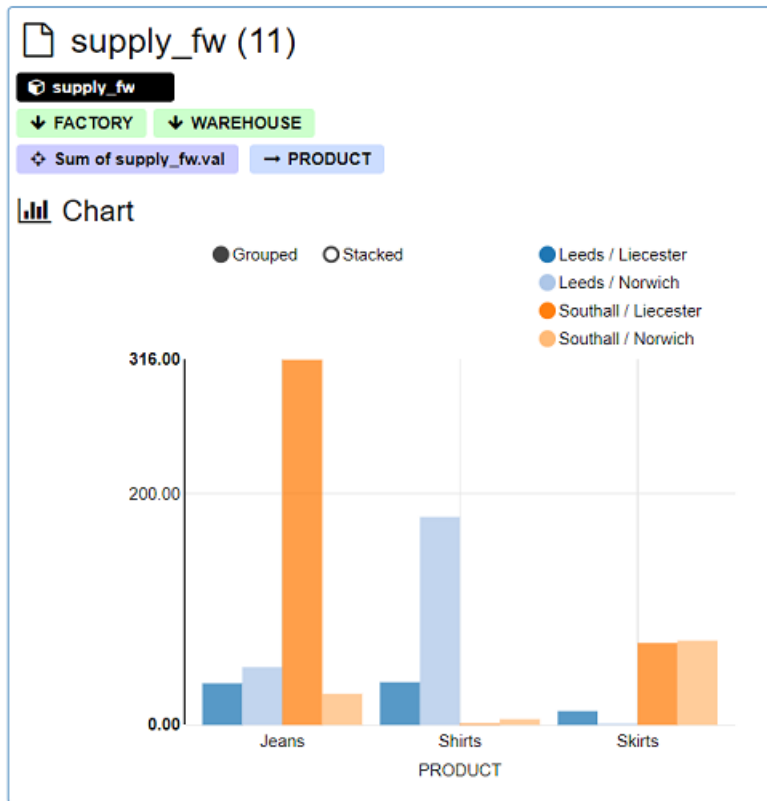


Figure 6.8: Supply from Factory to Warehouse [Drilled Down]

6.8. Dicing across the factory dimension will give us the data for individual factories. If we consider only one factory *e.g* *Southall*, then the resulting graph is shown in Figure 6.9. Similarly, a filter can be chosen across other dimensions as well.

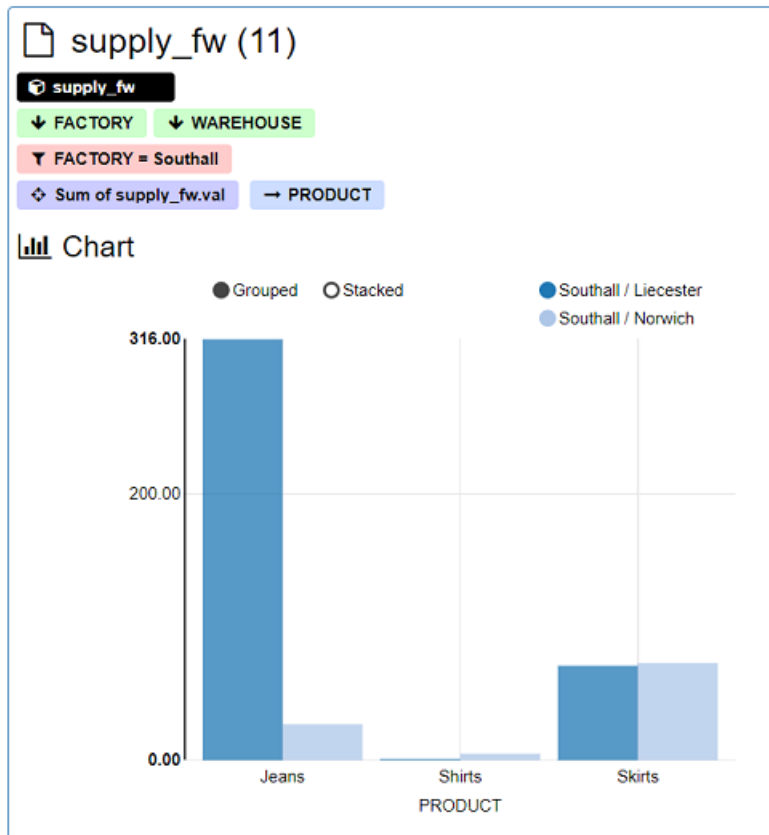


Figure 6.9: Supply from Factory to Warehouse [Drilled Down with filter]

6.4 A Perspective from a Business Owner

We have illustrated various examples of data visualization components. In this section, we give practical scenarios of business problems.

The problem owner wants to evaluate the production of garments in different factories. Depending on the capacity and actual production, the decision needs to be made, whether to expand operations or close down production at certain factories.

Let us assume the problem owner has access to all the data and the framework described in this research is used for the optimization model under consideration. A possible solution approach is given below

- Instantiation of the model from the dashboard. The loaded model can be solved with the default parameter set. Then the problem owner can explore the entities like the number of garments produced and production capacity. The data visualization component becomes extremely critical at this point.
- If closing down production is the best option, then the problem owner can create a realization of this scenario from the interactive input dashboard. This can be done by simple discarding (un-checking) the values for the factories. The change in result can be analysed by solving the updated model from the dashboard.
- In another scenario, the problem owner may decide to increase the production capacity. This realization can be achieved from the input dashboard as well. The input data for this can easily be modified in the tables for production capacity. The corresponding output can be analysed after solving the updated model.

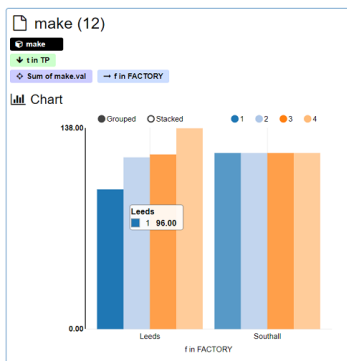


Figure 6.10: Production in Factories over Time Period

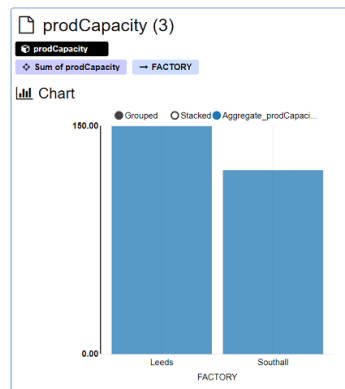


Figure 6.11: Production Capacity at Factories

6.4.1 Further Insight

Analysis of the data visualization component can give the problem owner the motivation to carry out different scenarios.

From Figure 6.10, we observe that one of the factories is producing at near full capacity and another one is producing well below its capacity. This might be a factor which can lead the problem owner to consider the idea of closing one of the factories. Once a factory is closed, the production capacity of the other factory is to be increased in order to meet the demand. This realization can be achieved by the following methods.

Case 1: From the interactive dashboard, the decision maker can easily make the selection of factories as input parameters. Furthermore, the capacity table and production rate can be modified from the data-table part of the dashboard. The corresponding solution can be analysed further with the modified results.

Case 2: However, the practical scenario is more complex than the above case. There is usually a cost associated with the opening of a factory. Similarly closing a factory comes at a cost. To incorporate this, the underlying decision model needs to be modified. It becomes the responsibility of the model expert to introduce a variable which can express the opening and closing status of the factory. This is presented in (14).

$$y_f = \begin{cases} 1, & \text{if } open \\ 0, & \text{if } closed \end{cases} \quad \forall f \quad (14)$$

Similarly, a variable can be introduced to represent the opening and closing status of the warehouse. This is expressed in (15).

$$z_w = \begin{cases} 1, & \text{if } open \\ 0, & \text{if } closed \end{cases} \quad \forall w \quad (15)$$

Expansion of capacity is required in some of the factories and warehouses in order to meet the requirement, when some factories are closed. For this, a few more variables are introduced as given in (16).

$$y'_f = \begin{cases} 1, & \text{if expansion required} \\ 0, & \text{otherwise} \end{cases} \quad \forall f \quad (16)$$

The cost factor due to the closure of factories and the increase in capacity needs to be considered as part of the total expense. These cost components are expressed in equation (17) as shown below.

$$Cost_{total} = Cost_{closing} + Cost_{capacity_expansion}$$

$$Cost_{tot} = \sum_{f \in F} CC_f * (1 - y_f) + \sum_{f \in F} CX_f * y'_f \quad (17)$$

Similarly, all capacity constraints have to be reformulated using these new variables.

6.4.2 Mixed Integer Problem

Introducing the parameters explained in the previous section will convert the problem into a mixed integer problem (MIP). Some of the decision variables, in this case, are constrained to take integer values. The AMPL model can be modified according to the algebraic representation given in the previous section.

For the original LP model we have used the CPLEX solver. This solver can easily deal with the modified version of the model. However the performance and the solution method for both cases are different and out of the scope of this case study.

6.5 Summary of BI platform

Recent times have witnessed a rapid growth in Business Intelligence systems and applications in DSS. A generic approach to integrate a business intelligence system to optimization based DSS platform is discussed in this chapter. This brings together an optimization-based DSS and a BI system. It also comes with a unique interactive feature, which gives the business owner the freedom to create different business scenarios by updating the interactive web-based dashboard. The BI framework also provides embedded data visualization components for results based on different inputs. The function of this framework acts as a feedback loop as given in Figure 6.12.

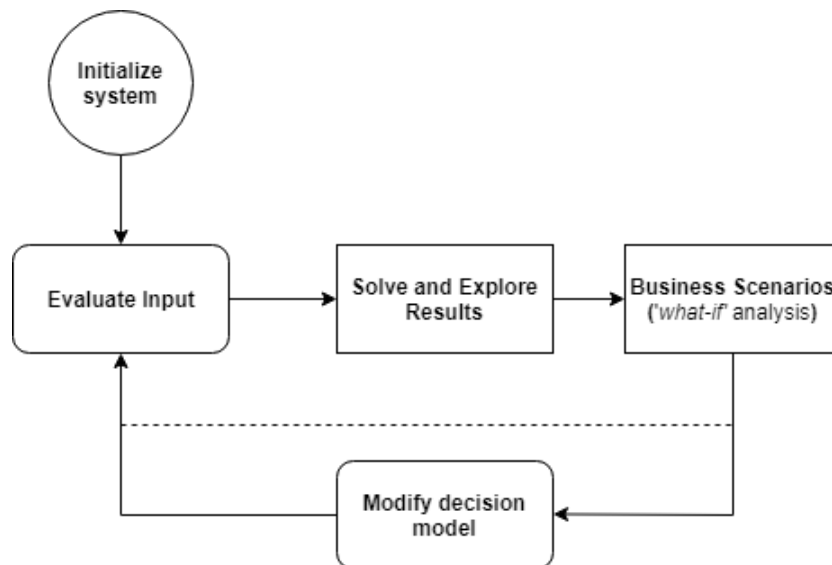


Figure 6.12: BI System as Feedback Loop

7 Use of the Framework in other Domains

The objective of this research is to have a generic framework which is applicable to diverse domains. The concept and structure of the framework components should remain unchanged when there is a change in the decision model. This nature is verified with some of the well-known optimization problems like cutting stock problem, diet problem and portfolio optimization. In this chapter, however, only one problem is presented in detail which is an Asset Liability Management (ALM) model taken from the domain of financial analytics. Other problems follow a similar approach and are not covered.

7.1 The ALM Problem

An investment manager in charge of a ‘Pension Scheme’ faces the problem of creating a portfolio by initially allocating assets and then rebalancing the portfolio by buying and selling the assets at subsequent time periods. He also needs to consider future obligations (liabilities) which the pension scheme must satisfy. Typical examples of pension scheme liabilities are (a) making regular payments to the retirees, (b) making pay-outs to spouses in the case of the death of scheme members and (c) drawdown for education or health-related borrowings. In this illustrative example, the assets are chosen out of three asset classes, equities, bonds and commodities. The goal of the investor is to maximize the portfolio wealth at the end of a predefined time horizon. There are further restrictions that at least some percentage must be invested in bonds and at most a certain percentage may be invested in equities. In each period of the time horizon, and for each asset considered, the investor needs to decide the amount of each asset to buy, sell and hold.

7.2 Algebraic Model

Model sets, parameters and variables are in Tables 7.1, 7.2 and 7.3 below.

Notation	Description
A	Set of assets
T	Sequential time period
S	Set of sectors
C	Set of asset class
AC_c	Set of assets belonging to asset class c
AS_s	Set of assets belonging to sector s

Table 7.1: List of Sets

7.2.1 Objective Function

The objective is to maximize the portfolio value at the end of the time horizon (NT). This is expressed in (18) as shown below.

$$\text{maximize } MV_{NT} \quad (18)$$

7.2.2 Constraints

Asset Holding Constraints: During the planning horizon, the portfolio is re-balanced at discrete points in time (beginning of each time period). The model gives an estimation of buying or selling assets in each period while maintaining the restriction proposed in the constraints. The asset holding constraint shows the portfolio composition over time.

$$H_{a1} = I_a + B_{a1} - S_{a1} \quad a \in A \quad (19)$$

Notation	Description
UC_c	Upper limit for asset class c
LC_c	Lower limit for asset class c
UCS_{sc}	Upper limit of asset sector s in asset class c
LCS_{sc}	Lower limit of asset sector s in asset class c
P_{at}	Price of a at time t
CI_t	Cash inflow at time t
I_a	Initial holding of asset a
L_t	Liability at time t
g	Transaction cost percentage
ts	Total transaction cost for selling $1 - g$
tb	Total transaction cost for buying $1 + g$

Table 7.2: List of Parameters

Notation	Description
H_{at}	Hold asset a in time t
S_{at}	Sell asset a in time t
B_{at}	Buy asset a in time t
MC_{ct}	Market value of asset class c time t
MV_t	Total market value at the end of time t

Table 7.3: List of Variables

$$H_{at} = H_{at-1} + B_{at} - S_{at} \quad a \in A, t \in 2, \dots, NT \quad (20)$$

Fund Balance Constraints: The buying of assets and liabilities lead to cash outflows which should be equal to cash inflows which arise due to selling of some of the assets of potential increased returns. The constraint set out below represents this fund balance of the portfolio over time.

$$\left(\sum_{a \in A} S_{a,t} * P_{at} \right) * ts - L_a + CI_t = \left(\sum_{a \in A} B_{a,t} * P_{at} \right) * tb \quad t \in T \quad (21)$$

Market Value Constraints: We use constraints to define the dependency between the accounting variables and the decision variables. The partial mark to market values are defined as follows

$$MC_{ct} = \sum_{a \in AC_c} H_{a,t} * P_{at} \quad \begin{array}{l} t \in T, c \in C \\ LC_c > 0 \text{ or } UC_c < 100 \end{array} \quad (22)$$

The total market value of the investments is thus simply,

$$MV_t = \sum_{a \in A} H_{a,t} * P_{at} \quad t \in T \quad (23)$$

Asset Class Constraints: The following constraint limits the maximum exposure to a defined percentage of the mark to market value in all time buckets. Also, it ensures that at least a defined percentage of the mark to market value is invested in all time buckets is obeyed.

$$MC_{ct} \leq UC_c * MV_t \quad \begin{array}{l} t \in T, c \in C \\ UC_c < 100 \end{array} \quad (24)$$

$$MC_{ct} \geq LC_c * MV_t \quad \begin{array}{l} t \in T, c \in C \\ LC_c > 0 \end{array} \quad (25)$$

Asset Sector Constraints: Each asset class consists of a series of asset sectors. The following constraints limit the maximum exposure to be a defined percentage of a sector in an asset class. Also, it ensures that at least the defined percentage of a sector in an asset class invested in all time buckets.

$$LCS_{cs} \leq \sum_{a \in AC_c \cap AS_s} H_{at} * P_{at} \quad \begin{array}{l} t \in T, c \in C, s \in S \\ LCS_{cs} > 0 \end{array} \quad (26)$$

$$UCS_{cs} \geq \sum_{a \in AC_c \cap AS_s} H_{at} * P_{at} \quad \begin{array}{l} t \in T, c \in C, s \in S \\ UCS_{cs} < 100 \end{array} \quad (27)$$

7.3 Model Formulation and Result

The model is formulated in AMPL. The model file and data files are given in Appendix C. Once the model formulation is done, we instantiate it using a default dataset. The underlying CPLEX solver is hooked to the AMPL system for solving the model.

Input Data

In this example, we have used a small data set which includes a few assets. The entire data file contents are given in section C.2 and C.3 of Appendix C. Some part of the data is stored in a relational database management system which is easily accessible via AMPL commands. Tables A.9 and A.10 of Appendix A represents the data for asset details and asset prices respectively. Assets are further classified based on asset classes and sectors. Table A.8 presents these details.

7.4 Dashboard for the Optimization Model

As given in Appendix C, we have a separate decision model and data file. The data file is created by using the master set of coefficients which can be used for a decision model. The decision maker has access to all input parameters during the investigation of the problem. Thus, every possible value is included in the input. Also, the responsibility of the model expert is to design the model in such a way that any change in configuration parameter does not require alteration to the model file. The underlying framework can produce a modified version of input data. These data and configurations can be exploited fully if the master data set includes all the possible values. The API used in this framework is responsible for reading and parsing of the mathematical model written in AMPL. Loading the model components to the interactive dashboard and interaction of the AMPL system to the solver system is also carried out by the API. A complete overview of the dashboard is shown in Figure 7.1.

Instantiation of the System: In order to make the entire system work, a configuration file needs to be provided with information regarding the model file, data file and other related files. A sample configuration is shown in Figure A.1 under section A.2 of Appendix A. A configuration file with all valid entries is enough to load the dashboard. Clicking on ‘Load Model’ instantiates the model and populates the dashboard with the master dataset for the ALM problem.

Running and Solving: A series of commands for solving a model is given in section 4.4.2. Any conditional statement, reassignment and other control statements can be combined together and included in a single script file. A

Asset Liability Management

AMPL DIR - MODEL - DATA -
 D:\Workspace\Research\modelview\src\main\resources\ampl\script\

ASSETS	TP	asset	assetBySector
<input checked="" type="checkbox"/> ABEV3	<input checked="" type="checkbox"/> 1	<input checked="" type="checkbox"/> ABEV3	<input checked="" type="checkbox"/> BRAZIL10YEAR
<input checked="" type="checkbox"/> BRAZIL10YEAR	<input checked="" type="checkbox"/> 2	<input checked="" type="checkbox"/> BVSP	<input checked="" type="checkbox"/> BRAZIL1YEAR
<input checked="" type="checkbox"/> BRAZIL1YEAR	<input checked="" type="checkbox"/> 3	<input checked="" type="checkbox"/> ITSA4	<input checked="" type="checkbox"/> CORN
<input checked="" type="checkbox"/> BVSP	<input checked="" type="checkbox"/> 4	<input checked="" type="checkbox"/> PETR4	<input checked="" type="checkbox"/> OIL
<input checked="" type="checkbox"/> CORN		<input checked="" type="checkbox"/> USIM5	<input checked="" type="checkbox"/> US10YEAR
<input checked="" type="checkbox"/> ITSA4		<input checked="" type="checkbox"/> VALE5	
<input checked="" type="checkbox"/> OIL			assetclasses
<input checked="" type="checkbox"/> PETR4			<input checked="" type="checkbox"/> BN

	BN	60
--	----	----

param minsector(assetclasses, sectors) default 0;

param price{ASSETS, TP};

index0	index1	price
ABEV3	1	4.63
ABEV3	2	4.57

Decision Variables	Objective
	maximize wealth: MarketValuetot{NT};

Load Model
Solve Model

Success : [true] Explore AMPL Entities

Solver Message : CPLEX 12.6.3.0: optimal solution; objective 158507324 50 dual simplex iterations (34 in phase I)

[Cubes Server Info](#)

[AMPL Entities List \(Cubes\)](#)

Figure 7.1: ALM Dashboard

response from the solver for the ALM model is given below.

The solver output :

```
CPLEX 12.6.0.0: optimal solution;  
objective 158507324  
50 dual simplex iterations (34 in phase I)
```

7.5 Further Analysis

Solving the model with a series of AMPL commands is not complex for the model expert. However to make the dashboard easy for the decision owner, a different set of controls are provided to initiate the solving process. These are helpful, if the decision maker decides to solve different instances of the model. Intervention from the model expert is only required when the changes are required to be incorporated in the decision model.

Assets are listed in the set section of the dashboard. Let us assume the decision maker decides to exclude ‘*ABEV3*’ from the portfolio. There are two ways it can be achieved as discussed below,

Change in the Decision Model/Data:

The data table needs to be updated with a new set of assets and all the data references need to be removed which include ‘*ABEV3*’ as an index parameter. Alternatively, the AMPL script by the model expert needs to be modified for the set data before solving the model. Once these changes are done a list of AMPL command gives the below result.

```
CPLEX 12.6.0.0: optimal solution;  
objective 156815464.6
```


45 dual simplex iterations (31 in phase I)

Change in Interactive Dashboard:

From the interactive dashboard, the decision maker can easily make the selection of ‘Assets’ as input parameters. Figure 7.2 shows the example of the changed input and the corresponding output. Furthermore other operations like changing the parameter table, selecting configuration options and limits can be done from the dashboard as well.

The screenshot displays the 'Interactive Dashboard' interface. It features two columns of selection options: 'rASSETS' and 'sectors'. The 'rASSETS' column includes checkboxes for ABEV3, BRAZIL10YEAR, BRAZIL1YEAR, BVSP, CORN, ITSA4, OIL, and PETR4. The 'sectors' column includes checkboxes for BAM, COS, FNS, and NA. A large black arrow points to the ABEV3 checkbox in the 'rASSETS' column. Below the selection options, a status bar shows 'Success : [true]' and a blue button labeled 'Explore AMPL Entities'. At the bottom, a 'Solver Message' box contains the text: 'CPLEX 12.6.3.0: optimal solution; objective 156815464.6 48 dual simplex iterations (33 in phase I)'. The numerical value '156815464.6 48' is highlighted with a black rectangular box.

Figure 7.2: Interactive Dashboard

Data Analysis

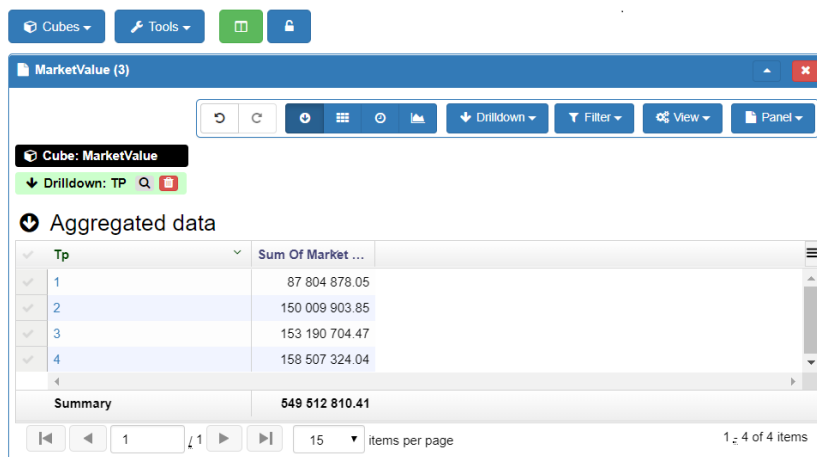
For the data analysis the *Cubes* framework is closely coupled with the interactive dashboard. The associated model input data and results are available as data cubes for further analysis. The OLAP cubes support basic

features like aggregated browsing, drilling down, slicing and dicing.

7.6 Analyst BI Platform

The BI tool interacts with the OLAP framework that handles all the complex queries and provides responses very quickly. Section 5.1 explains various OLAP features with examples.

Data Tables: If the drill-down parameter is selected as time period from the dashboard options, a data-table will be available as shown in Figure 7.3. The cube name and the drilled down dimension is displayed just above the table.



Aggregated data	Sum Of Market ...
1	87 804 878.05
2	150 009 903.85
3	153 190 704.47
4	158 507 324.04
Summary	549 512 810.41

Figure 7.3: *MarketValue* Data Drill Down over Time

Charts and Graphs: The control options in the BI platform gives the flexibility to view the data in the form of charts and graphs. Figure 7.4 shows the bar chart of *MarketValue* drilled down over time period with horizontal dimension selected as *assetclass*. This helps in identifying the market value

per asset class (bond, equity) in each period. The cube name, measure entity and horizontal dimension are displayed just above the bar chart.

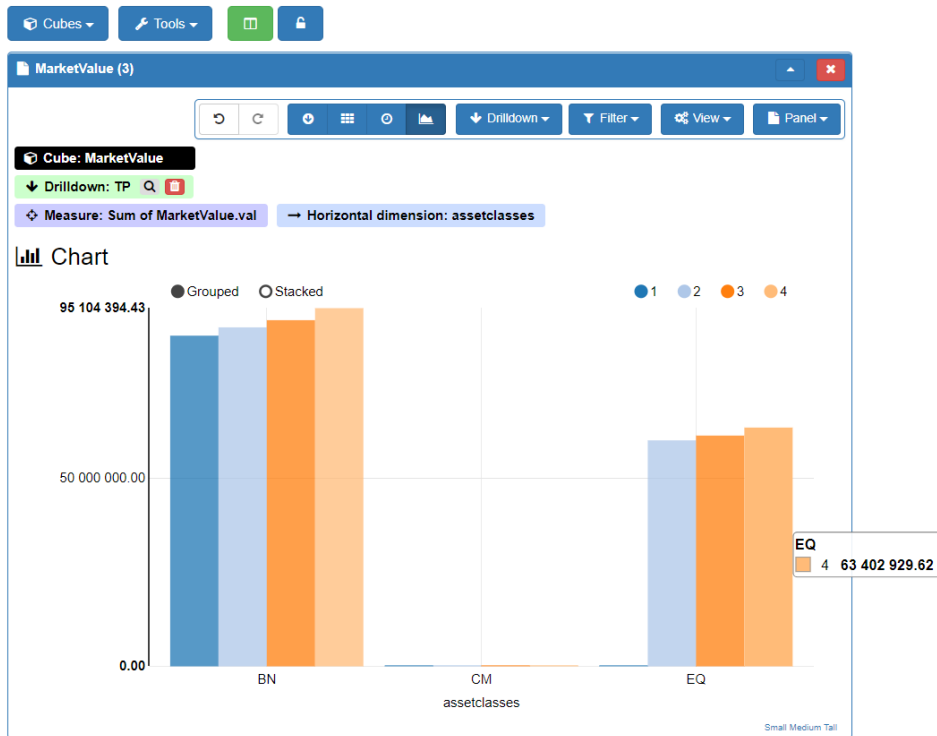


Figure 7.4: *MarketValue* Bar Chart Drill Down over Time

Filter Data: Drilled down data across the dimension gives us all the values from the result. In certain circumstances, the analyst may be interested in a smaller chunk of data. This can be achieved in the BI platform by choosing the filter option. Figure 7.5 shows the pie chart of *MarketValue* for a particular time period ($t = 2$). The horizontal dimension selected in this case is *assetclass*. The cube name, measure entity and horizontal dimension are displayed just above the bar chart.

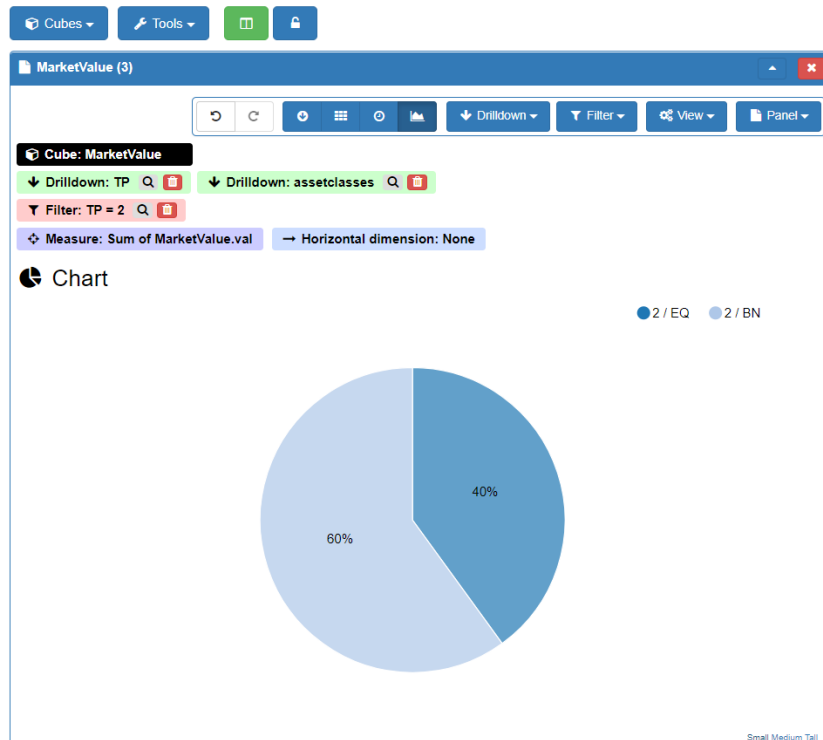


Figure 7.5: *MarketValue* Pie Chart Drill Down over Time

An Improved Dashboard

Upon close analysis of the model, we have identified certain parameters which are set at a maximum/minimum level to add restrictions to the portfolio. For this type of set up, more customization can be done to the input dashboard. Some improved UI controls like a slider and selection options can be added to the dashboard which is illustrated in Figure 7.6. However, these customizations are specific to the requirement of the problem under consideration and out of the scope of this research.

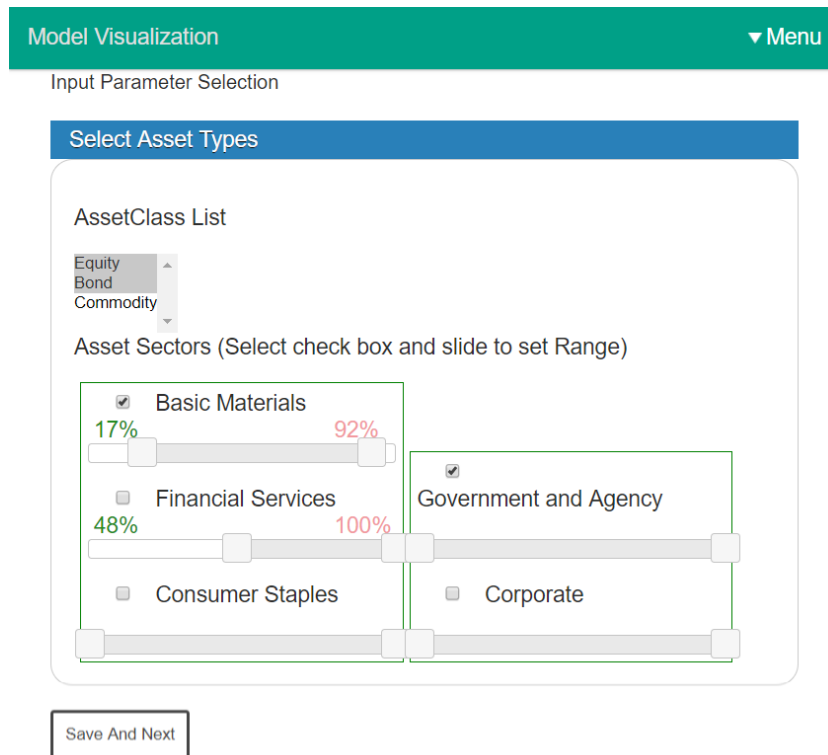


Figure 7.6: A Customized Version of ALM Dashboard

Summary

This chapter covers the application of the proposed framework to an asset liability management problem. An abstract data-driven model is created for the business problem. An API based solution from the web-based dashboard is presented. Finally, the BI platform is explained which is integrated into the dashboard. Analysis from the dashboard enables the decision maker to have better insight and understanding of the business processes. Features like association, summarization, aggregation can address the business questions, such as ‘what is the average amount spent on a particular asset class’, ‘In which time did the portfolio achieve its maximum value’.

Furthermore, the dashboard can be used by different users in a DSS. In this

ALM problem, individual traders can use the BI platform with the dashboard to analyse the portfolio. The process involves changing the input parameter from the dashboard and analysing the corresponding result (mark-to-market) values. However, the organisation regulatory bodies can use the dashboard in a different way. Certain limitations are enforced in the model for different asset classes (bonds, equity). The breakdown of assets by asset-classes and the market value of asset-classes can be analysed in the dashboard for compliance purposes.

8 Conclusion

8.1 Overview

Chapter 7 explains the use of the proposed interactive BI platform for an optimization-based decision problem involving asset and liability management. An abstract data-driven decision model is presented for the business problem. This is driven by a web-based dashboard built around an API based framework. Analysis and insight into the business problem are discussed with the integrated BI system which includes multidimensional data and results.

This chapter presents the summary of the thesis. Thereafter, the attainment of research aim and objectives are discussed. Subsequently, the major contributions of the thesis are highlighted and explained. Finally, a conclusive remark on this research is presented.

8.2 Summary

This thesis aimed to provide an interactive BI platform for generic optimization based DSS. The implementation started with a decision problem. We analysed the business problem and presented the optimization problem in algebraic form. In subsequent chapters, we created the optimization model for the problem. Then we introduced a framework, which is API driven and can be used to solve the model and collect the information about the underlying optimization system. The API retrieved information is presented in an interactive web-based platform. This platform is highly interactive. Therefore, all the users operating the platform can run, solve and simulate different model instances without core expertise on model and data components.

Upon analysis of the data associated with the input model and output results, we established its similarity to multidimensional data views and integration of OLAP tools of it. We demonstrated various OLAP features like drill down, roll up and aggregation. Finally, we introduced the data visualization components and its use in OLAP databases. These components are also part of the proposed framework, which is an easy to use end-to-end platform for a problem owner. We summarize our findings as,

1. *Flexible and abstract decision model*: It is the responsibility of the model expert to design a generic model for the decision problem under investigation. Also, the model should be flexible enough in order to incorporate different business scenarios by changing data and configuration.
2. *Data-driven components*: The database expert plays a key role in making the components of the model data driven. This abstraction can be achieved by analysing input data and configuration parameters. A master dataset with all feasible input is essential as it helps to create different subsets (coefficients) for different scenarios.
3. *Dashboard generation*: Using the software tool a dashboard gets generated. The interactive dashboard comes with BI capabilities. The problem owner can benefit from the features of the dashboard while doing analysis and making business decisions.
4. *Further insight into DSS*: The analysis using the interactive dashboard and BI platform gives the problem owner motivation to carry out different business scenarios. This may lead to further enhancement in the decision model. For example, binary and integer variables or

parameters with uncertainty and scenarios can be introduced to the decision model and the dashboard.

8.3 Research Aim and Objectives

The aim of this research is to provide an interactive BI platform for generic optimization based DSS. The concepts and frameworks are illustrated with a supply chain management problem. Six critical objectives are considered in order to achieve the goal of this thesis. An overview of the accomplished objectives are outlined below:

Objective 1: The first objective was to set out and understand optimization based decision problems for defined domains. This is achieved in chapter 2 where two different decision problems are discussed. The supply chain problem is discussed in detail along with its algebraic formulation.

Objective 2: The second objective was to formulate an abstract data-driven optimization model based on the algebraic model. This is achieved using algebraic modeling language tools and solvers. Chapter 3 defines an abstract data-driven optimization model in AMPL. Different entities in AMPL are expressed in the form of abstract modeling entities like sets, indices, parameters, variables and constraints.

Objective 3: The third objective was to design an interactive dashboard to analyse and solve the optimization model. A web-based dashboard is designed for this purpose as given in chapter 4. The dashboard is created using HTML 5, CSS and JavaScript. The back-end system consists of a python API in AMPL and a python-based web framework. Instantiation of the system, running and solving different instances of

the underlying decision problem based on user interactions are explained in this chapter.

Objective 4: The fourth objective was to provide an analysis method for multidimensional input data and results associated with the optimization model. *Cubes* OLAP framework is used for this purpose which converts optimization system data to logical models consisting of data-cubes. The logical model is used for analysis in terms of facts, dimensions and hierarchies. This is explained in chapter 5.

Objective 5: The fifth objective was to have an integrated BI system with the web-based dashboard. To achieve this, the *Cubesviewer* library is integrated into the web-based dashboard. It supports data exploration, reporting and data visualization from various input sources as well as from the instances of the model created by the dashboard. Various examples are illustrated in chapter 6.

Objective 6: The final objective was to demonstrate the generic nature of the proposed platform with similar optimization-based decision problems from other domains. The concepts and proposed framework have been explained from chapter 3 to 6 with a well-established supply chain management problem. Chapter 7 applies the same ideas to an ALM problem from the finance domain.

8.4 Contributions

Some of the major contributions of this thesis are,

- (i) *Single dashboard for DSS users:* Multiple instances of a model are solved from a simple web-based dashboard. Hence a quick turn around time

for solving a business scenario. This may give the problem owner the motivation to carry out a different scenario and identify the changes that require modifications from model and DB experts. This ensures interaction between different DSS users is well defined.

(ii) *Analytic platform for the Optimization system*: A framework bundles the analytic platform to the optimization system. This is applicable for complex models with a large dataset and scenarios. The OLAP platform performs MDX queries directly from the Optimization system results (Input). The platform is web-based. Hence it is highly accessible and can be deployed in a client-server architecture.

(iii) *The generic use of proposed framework*: This is applicable for optimization models for diverse domains with a certain level of abstraction. Some of the other examples which are not covered in this write-up are the cutting stock problem, and diet problem. These have been created using the proposed platform. Therefore a rapid prototype version of an interactive web-based dashboard can be made possible from any optimization model.

8.5 Limitations and Future Directions

One limitation of the proposed platform is that it is applicable for optimization models that provides control over model components and configuration from associated data. In the context of optimization model, the platform can only make re-use of the existing model and its instances. Creating a user-defined model and adding entities to the existing model is not supported via web-based dashboard. Another limitation is the specific selection of algebraic modeling language tools and BI framework. The platform is built

with AMPL, related API and other python-based frameworks. This type of set up may not be achieved with different modeling language tools.

This research can be extended to a number of future projects. The ideas of this research can be applied to improve the analytical ability of any optimization-based system. Some of the state-of-the-art AML systems can be extended to support basic BI related queries. This may include built-in features to produce visualization components for aggregated, filtered and drilled-down data. Another future project can be performance testing of various linear and non-linear solvers for a web-based optimization dashboard. Further development may look into a novel approach for embedding large scale mathematical models to web-based optimization dashboard and make the components accessible as per user role. The problem owner can access the dashboard with interactive BI tools. The dashboard for model expert will be different focusing on optimization model components. Similarly, the dashboard for database expert will only represent the logical data structure associated with the model components.

8.6 Concluding Remarks

The outcome of this research provides a way to link the optimization system and BI system together. In this research, a sufficiently generic framework and concept is explained to create an interactive BI platform for optimization based DSS. This platform can be used to perform optimization related tasks without core knowledge of the underlying modeling components. At the same time, the integrated BI system can be used for analysis and reporting. These are equally important in the decision-making process. We have also emphasized the ability of the system to analyse the output which in turn can give motivation for new input for different realizations of the problem. This

feedback mechanism provides an advantage to the business owner and limits the role of a model specialist. These ideas and methodologies can be used together to create valuable decision support tools in every domain.

References

- AMPL API* (n.d.). <https://ampl.com/products/api/>. Accessed: 2018-06-01.
- Bharati, Pratyush and Abhijit Chaudhury (2004). “An empirical investigation of decision-making satisfaction in web-based decision support systems”. In: *Decision Support Systems* 37.2, pp. 187–197. ISSN: 0167-9236. DOI: [https://doi.org/10.1016/S0167-9236\(03\)00006-X](https://doi.org/10.1016/S0167-9236(03)00006-X). URL: <http://www.sciencedirect.com/science/article/pii/S016792360300006X>.
- Bidhandi, Hadi Mohammadi, Rosnah Mohd. Yusuff, Megat Mohamad Hamdan Megat Ahmad, and Mohd Rizam Abu Bakar (2009). “Development of a new approach for deterministic supply chain network design”. In: *European Journal of Operational Research* 198.1, pp. 121–128. ISSN: 0377-2217. DOI: <https://doi.org/10.1016/j.ejor.2008.07.034>. URL: <http://www.sciencedirect.com/science/article/pii/S0377221708006735>.
- Borade, Atul B and Satish V Bansod (2007). “Domain of supply chain management-a state of art”. In: *Journal of Technology Management & Innovation* 2.4.
- Codd, E. F. (June 1970). “A Relational Model of Data for Large Shared Data Banks”. In: *Commun. ACM* 13.6, pp. 377–387. ISSN: 0001-0782. DOI: 10.1145/362384.362685. URL: <http://doi.acm.org/10.1145/362384.362685>.
- Codd, Edgar F (1979). “Extending the database relational model to capture more meaning”. In: *ACM Transactions on Database Systems (TODS)* 4.4, pp. 397–434.

- Codd, Edgar F (1990). *The relational model for database management: version 2*. Addison-Wesley Longman Publishing Co., Inc.
- Codd, Edgar F, Sharon B Codd, and Clynch T Salley (1993). “Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate”. In: *Codd and Date* 32.
- Crockford, Douglas (2009). “Introducing json”. In: *Available: json.org*.
- Cubes* (n.d.). <http://cubes.databrewery.org/>. [Online; accessed 18-June-2018].
- CubesViewer* (n.d.). <http://www.cubesviewer.com/>. [Online; accessed 18-June-2018].
- Das, Chandrasekhar and Rajesh Tyagi (1994). “WHOLESALE: A Decision Support System for Wholesale Procurement and Distribution”. In: *International Journal of Physical Distribution & Logistics Management* 24.10, pp. 4–12. DOI: 10.1108/09600039410074746.
- Date, Christopher J (1990). *An introduction to database systems. Volume 1*.
- Dengiz, Berna, Tolga Bektas, and A. Eren Ultanir (2006). “Simulation optimization based DSS application: A diamond tool production line in industry”. In: *Simulation Modelling Practice and Theory* 14.3, pp. 296–312. ISSN: 1569-190X. DOI: <https://doi.org/10.1016/j.simpat.2005.07.001>. URL: <http://www.sciencedirect.com/science/article/pii/S1569190X0500081X>.
- Desanctis, Gerardine and R Brent Gallupe (1987). “A foundation for the study of group decision support systems”. In: *Management science* 33.5, pp. 589–609.
- Dutta, Goutam, Robert Fourer, Akhilesh Majumdar, and Debabrata Dutta (2007). “An optimization-based decision support system for strategic

- planning in a process industry: The case of a pharmaceutical company in India”. In: *International Journal of Production Economics* 106.1. Special section on contextualisation of supply chain networks, pp. 92–103. ISSN: 0925-5273. DOI: <https://doi.org/10.1016/j.ijpe.2006.04.011>. URL: <http://www.sciencedirect.com/science/article/pii/S0925527306000831>.
- Fourer, Robert, David M. Gay, and Brian W. Kernighan (2002). *AMPL: A Modeling Language for Mathematical Programming*. Cengage Learning. ISBN: 0534388094.
- Gay, David M (1997). *Hooking your solver to AMPL*. Tech. rep. Citeseer.
- Geoffrion, A. and S. Maturana (1995). “Generating optimization-based decision support systems”. In: *Proceedings of the Twenty-Eighth Annual Hawaii International Conference on System Sciences*. Vol. 3, 439–448 vol.3. DOI: 10.1109/HICSS.1995.375626.
- Gray, Paul (1987). “Group decision support systems”. In: *Decision Support Systems* 3.3, pp. 233–242. ISSN: 0167-9236. DOI: [https://doi.org/10.1016/0167-9236\(87\)90178-3](https://doi.org/10.1016/0167-9236(87)90178-3). URL: <http://www.sciencedirect.com/science/article/pii/0167923687901783>.
- Holsapple, Clyde W and Andrew B Whinston (1988). “The environment approach to decision support”. In: *Mathematical models for decision support*. Springer, pp. 539–556.
- Kallrath, Josef (2004). *Modeling Languages in Mathematical Optimization*. Vol. 88. Springer Science & Business Media.
- Kimball, Ralph and Margy Ross (2011). *The data warehouse toolkit: the complete guide to dimensional modeling*. John Wiley & Sons.

- Koutsoukis, Nikitas-Spiros, Belen Dominguez-Ballesteros, Cormac A. Lucas, and Gautam Mitra (2000). “A prototype decision support system for strategic planning under uncertainty”. In: *International Journal of Physical Distribution & Logistics Management* 30.7/8, pp. 640–661. DOI: 10.1108/09600030010346387.
- Maniatis, Andreas S., Panos Vassiliadis, Spiros Skiadopoulos, and Yannis Vassiliou (2003). “Advanced Visualization for OLAP”. In: *Proceedings of the 6th ACM International Workshop on Data Warehousing and OLAP*. DOLAP '03. New Orleans, Louisiana, USA: ACM, pp. 9–16. ISBN: 1-58113-727-3. DOI: 10.1145/956060.956063. URL: <http://doi.acm.org/10.1145/956060.956063>.
- Mansmann, Svetlana and Marc H. Scholl (2007). “Exploring OLAP Aggregates with Hierarchical Visualization Techniques”. In: *Proceedings of the 2007 ACM Symposium on Applied Computing*. SAC '07. Seoul, Korea: ACM, pp. 1067–1073. ISBN: 1-59593-480-4. DOI: 10.1145/1244002.1244235. URL: <http://doi.acm.org/10.1145/1244002.1244235>.
- Meixell, Mary J. and Vidyaranya B. Gargeya (2005). “Global supply chain design: A literature review and critique”. In: *Transportation Research Part E: Logistics and Transportation Review* 41.6. Global Logistics, pp. 531–550. ISSN: 1366-5545. DOI: <https://doi.org/10.1016/j.tre.2005.06.003>. URL: <http://www.sciencedirect.com/science/article/pii/S1366554505000487>.
- Mitra, Gautam, Cormac Lucas, Shirley Moody, and Bjarni Kristjansson (1995). “Sets and indices in linear programming modelling and their integration with relational data models”. In: *Computational Optimization and Applications* 4.3, pp. 263–283.

- Olszak, Celina M and Ewa Ziemba (2007). “Approach to building and implementing business intelligence systems”. In: *Interdisciplinary Journal of Information, Knowledge, and Management* 2.1, pp. 135–148.
- Philippakis, A. S. (1988). “Structured what if analysis in DSS models”. In: *[1988] Proceedings of the Twenty-First Annual Hawaii International Conference on System Sciences. Volume III: Decision Support and Knowledge Based Systems Track*. Vol. 3, pp. 366–370. DOI: 10.1109/HICSS.1988.11929.
- Power, Daniel J (2002). *Decision support systems: concepts and resources for managers*. Greenwood Publishing Group.
- Power, Daniel and Shashidhar Kaparathi (Jan. 2001). “Building Web-based decision support systems”. In: 11.
- Python Programming Language* (n.d.). <http://www.python.org/>. [Online; accessed 18-June-2018].
- Riabacke, Ari, Aron Larsson, and Mats Danielson (2011). “Business Intelligence as Decision Support in Business Processes: An Empirical Investigation”. In: *Proceedings of 2nd International Conference on Information Management and Evaluation (ICIME), ACI*, pp. 384–392.
- Rowen, William, Il-Yeol Song, Carl Medsker, and Edward Ewen (2001). “An analysis of many-to-many relationships between fact and dimension tables in dimensional modeling”. In: *Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW 2001), Interlaken, Switzerland*.
- Shaw, Michael J., Pei-Lei Tu, and Prabuddha De (1988). “Applying machine learning to model management in decision support systems”. In: *Decision Support Systems* 4.3, pp. 285–305. ISSN: 0167-9236. DOI: [https://doi.org/10.1016/0167-9236\(88\)90017-6](https://doi.org/10.1016/0167-9236(88)90017-6). URL:

<http://www.sciencedirect.com/science/article/pii/S0167923688900176>.

Shim, J.P., Merrill Warkentin, James F. Courtney, Daniel J. Power, Ramesh Sharda, and Christer Carlsson (2002). "Past, present, and future of decision support technology". In: *Decision Support Systems* 33.2. Decision Support System: Directions for the Next Decade, pp. 111–126. ISSN: 0167-9236. DOI: [https://doi.org/10.1016/S0167-9236\(01\)00139-7](https://doi.org/10.1016/S0167-9236(01)00139-7). URL: <http://www.sciencedirect.com/science/article/pii/S0167923601001397>.

Techapichetvanich, Kesaraporn and Amitava Datta (2005). "Interactive Visualization for OLAP". In: *Computational Science and Its Applications – ICCSA 2005*. Ed. by Gervasi, Osvaldo, Gavrilova, Marina L., Kumar, Vipin, Laganà, Antonio, Lee, Heow Pueh, Mun, Youngsong, Taniar, David, and Tan, Chih Jeng Kenneth. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 206–214. ISBN: 978-3-540-32045-6.

Whinston, BA, WC Holsapple, and HR Bonczek (1981). *Foundations of Decision Support Systems*.

Yeoh, William and Andy Koronios (2010). "Critical success factors for business intelligence systems". In: *Journal of computer information systems* 50.3, pp. 23–32.

Appendix A Illustrative Data and Configurations

A.1 Input Data Tables

All parameter values related to capacity (Demand, Penalty, Inventory Capacity, Warehouse Capacity, Initial Inventory, Initial Warehouse) present the number of units. Cost parameter values are in pounds per unit (£/unit).

Factory	Product	Inventory Capacity	Cost	Initial Inventory
Southall	Skirts	25	0.4	0
Southall	Shirts	0	0	5
Southall	Jeans	60	0.3	3
Leeds	Skirts	30	0.5	5
Leeds	Shirts	25	0.4	7
Leeds	Jeans	25	0.3	5

Table A.1: Inventory Details

Warehouse	Product	Warehouse Capacity	Cost	Initial Warehouse
Leicester	Skirts	25	0.4	1.0
Leicester	Shirts	0	0	2.2
Leicester	Jeans	9	0.3	2.6
Norwich	Skirts	30	0.5	0.6
Norwich	Shirts	25	0.4	1.8
Norwich	Jeans	25	0.3	1.2

Table A.2: Warehouse Details

Warehouse	Dealer	Transportation Cost
Norwich	London	0.3
Norwich	Paris	0.9
Norwich	Wien	1.1
Leicester	London	0.5
Leicester	Paris	1.1
Leicester	Wien	1.3

Table A.3: Transportation Cost (Warehouse to Dealer)

Factory	Dealer	Transportation Cost
Southall	London	1.0
Southall	Paris	2.2
Southall	Wien	2.6
Leeds	London	0.6
Leeds	Paris	1.8
Leeds	Wien	1.2

Table A.4: Transportation Cost (Factory to Dealer)

Factory	Warehouse	Transportation Cost
Southall	Leicester	0.1
Southall	Norwich	0.3
Leeds	Leicester	0.1
Leeds	Norwich	0.3

Table A.5: Transportation Cost to Warehouse

Dealer	Penalty		
	Skirts	Shirts	Jeans
London	7	10	17
Paris	7	10	17
Wien	7	10	17

Table A.6: Shortfall Penalty

Time	Dealer	Demand		
		Skirts	Shirts	Jeans
1	London	16	50	20
2	London	17	55	55
3	London	21	66	66
4	London	24	76	76
1	Paris	16	10	60
2	Paris	17	11	66
3	Paris	21	13	79
4	Paris	24	15	91
1	Wien	20	30	30
2	Wien	22	33	33
3	Wien	24	39	39
4	Wien	30	46	40

Table A.7: Dealer Demand

A.2 Configuration file for Dashboard

Sample configuration file for the framework is given below. It contains various entries like AMPL model and data file locations. Different entries of the file includes

- *AMPL_DIR*: This represents the directory to be considered as the working directory for the AMPL Process.
- *MODEL_DIR*: The location for the model file.
- *MODELS*: Model file name. Multiple file names can be included with some delimiter (,) separated.
- *DATA_DIR*: The location for the data file.
- *DATA* : Data file name. Multiple file-names can be included with comma separated values.
- *SCRIPTS* : Optional configuration. Can be used if the model and data components are loaded by an external script.

```
#-----AMPL_Dir-----  
AMPL_DIR = "C:\\Users\\Ansuman\\Desktop\\Ampl\\ampldev\\SPLC_V2\\script\"  
  
#-----MODEL-----  
MODEL_DIR = "C:\\Users\\Ansuman\\Desktop\\Ampl\\ampldev\\SPLC_V2\\model\"  
MODELS = "splc2.mod"  
  
#-----DATA-----  
DATA_DIR = "C:\\Users\\Ansuman\\Desktop\\Ampl\\ampldev\\SPLC_V2\\data\"  
DATA = "splc2.dat"  
  
#-----SCRIPTS-----  
#SCRIPTS = "splc3.run"
```

Figure A.1: Configuration File

A.3 Data Tables : Asset Liability Management

Code	Name	Asset Class	Class Name
COS	Consumer Staples	EQ	Equity
FNS	Financial Services	EQ	Equity
BAM	Basic Materials	EQ	Equity
COB	Corporate Staples	BN	Bond
GAB	Government and Agency	BN	Bond

Table A.8: Asset Sector Details

Asset	Asset Type	Asset Sector	Sector Name
BRAZIL10YEAR	BN	NA	NA
BRAZIL1YEAR	BN	NA	NA
US10YEAR	BN	NA	NA
CORN	CM	NA	NA
OIL	CM	NA	NA
ABEV3	EQ	COS	Consumer Staples
BVSP	EQ	FNS	Financial Services
ITSA4	EQ	FNS	Financial Services
PETR4	EQ	BAM	Basic Materials
USIM5	EQ	BAM	Basic Materials
VALE5	EQ	BAM	Basic Materials

Table A.9: Data Table for Assets

Asset	Price (T1)	Price (T2)	Price (T3)	Price (T4)
BRAZIL10YEAR	12.44	12.49	12.44	12.41
BRAZIL1YEAR	12.45	12.67	12.34	12.53
US10YEAR	3.37	3.42	3.47	3.29
CORN	659.5	722.5	693.25	754
OIL	101.01	111.8	117.36	125.89
ABEV3	4.63	4.57	5.1	5.54
BVSP	66575	67383	68587	66133
ITSA4	6.06	6.24	6.59	6.26
PETR4	24.76	26.13	26.06	23.52
USIM5	18.93	18.51	20.08	15.85
VALE5	39.68	38.59	36.88	35.85

Table A.10: Asset Price

Appendix B AMPL Files

B.1 Model File : Supply Chain Management

```
#Sets in Supply chain Problem

set FACTORY;

set PRODUCT;

set DEALER;

set WAREHOUSE;

param NT > 0;

set TP :=1..NT;

#Parameters in Supply chain Problem

param prodCost {PRODUCT,FACTORY};

param invCost{PRODUCT,FACTORY};

param inv_warehouse_Cost{PRODUCT,WAREHOUSE};

param trans_fw {FACTORY,WAREHOUSE} >= 0;

param trans_wd {WAREHOUSE,DEALER} >= 0;

param trans_fd {FACTORY,DEALER} >= 0;

param maxcost_fw =

max {f in FACTORY,w in WAREHOUSE} trans_fw[f,w];

param maxcost_wd =

max {w in WAREHOUSE,d in DEALER} trans_wd[w,d];

param maxcost_fd =

max {f in FACTORY,d in DEALER} trans_fd[f,d];
```

```

param penalty {PRODUCT,DEALER};
param invCapacity {PRODUCT,FACTORY};
param warehouseCapacity {PRODUCT,WAREHOUSE};
param initInv{PRODUCT,FACTORY}; #Initially at t=0
param init_invwarehouse{PRODUCT,WAREHOUSE};
param prodCapacity {PRODUCT,FACTORY};
param dealerdemand {PRODUCT,DEALER,TP}>=0;

var make{p in rPRODUCT,f in rFACTORY,t in TP}
>=0,<=prodCapacity[p,f];
var supply_fd {rPRODUCT,rFACTORY,rDEALER,TP :
maxcost_fd > 0} >=0;
var supply_fw {rPRODUCT,rFACTORY,rWAREHOUSE,TP :
maxcost_fw > 0} >= 0;
var supply_wd {rPRODUCT,rWAREHOUSE,rDEALER,TP :
maxcost_wd > 0} >= 0;
var inventory {p in rPRODUCT,f in rFACTORY,t in TP}
>=0,<=invCapacity [p,f];
var shortfall {rPRODUCT,rDEALER,TP}>=0;
var inv_warehouse {p in rPRODUCT,w in rWAREHOUSE,
t in TP : maxcost_fw > 0}
>= 0,<= warehouseCapacity[p,w];

var costTot >= 0 ;

```

```

#Objective Function
minimize cost_To_Mnfc: costTot;

subject to

#Total Cost = Prod Cos t + Trans Cost (FW- FD -WD)
+ Inv Cost + Penalty Cost
Total_cost:

costTot =
    (sum {p in rPRODUCT, f in rFACTORY, t in TP}
    prodCost [p,f] * make [p,f,t] ) +

    (sum {p in rPRODUCT, f in rFACTORY, w in rWAREHOUSE,
    t in TP : maxcost_fw > 0}
    trans_fw[f,w] * supply_fw [p,f,w,t])+

    (sum {p in rPRODUCT, f in rFACTORY, d in rDEALER,
    t in TP : maxcost_fd > 0}
    trans_fd[f,d] * supply_fd [p,f,d,t] ) +

    (sum {p in rPRODUCT,w in rWAREHOUSE, d in rDEALER,
    t in TP : maxcost_wd > 0}
    trans_wd[w,d] * supply_wd[p,w,d,t])
    +

    (sum {p in rPRODUCT, f in rFACTORY, t in TP}

```

```

    invCost[p,f]* inventory[p,f,t] ) +
    (sum {p in rPRODUCT, d in rDEALER, t in TP}
    penalty[p,d] * shortfall[p,d,t] ) +

    (sum {p in rPRODUCT, w in rWAREHOUSE,
    t in TP : maxcost_fw > 0}
    inv_warehouse_Cost[p,w]* inv_warehouse[p,w,t] );

C_TotalSupply {p in rPRODUCT,d in rDEALER, t in TP }:
    sum{ f in rFACTORY : maxcost_fd > 0}
    supply_fd [p,f,d,t]+
    sum{ w in rWAREHOUSE : maxcost_wd > 0}
    supply_wd [p,w,d,t]
    + shortfall[p,d,t] =
    dealerdemand[p,d,t];

C_TotalAtWareHouse_t1 {p in rPRODUCT,w in rWAREHOUSE}:
    sum{ f in rFACTORY : maxcost_fw > 0}
    supply_fw [p,f,w,1]
    + sum{P in rPRODUCT: maxcost_fw > 0}
    init_invwarehouse[p,w]
    = sum{ d in rDEALER : maxcost_wd > 0}
    supply_wd [p,w,d,1]
    +sum{P in rPRODUCT: maxcost_fw > 0}

```

```
inv_warehouse[p,w,1];
```

```
C_TotalAtWareHouse_t2 {p in rPRODUCT,w in rWAREHOUSE,  
t in 2..NT}:
```

```
  sum{ f in rFACTORY : maxcost_fw > 0}  
  supply_fw [p,f,w,t]  
+ sum{P in rPRODUCT: maxcost_fw > 0}  
  inv_warehouse[p,w,t-1]  
= sum{ d in rDEALER : maxcost_wd > 0}  
  supply_wd [p,w,d,t]  
+ sum{P in rPRODUCT: maxcost_fw > 0}  
  inv_warehouse[p,w,t];
```

```
C_InventoryBalanceInitial {p in rPRODUCT,  
f in rFACTORY}:
```

```
  initInv[p,f] + make[p,f,1] =  
  sum { d in rDEALER : maxcost_fd > 0}  
  supply_fd[p,f,d,1] +  
  sum {w in rWAREHOUSE: maxcost_fw > 0}  
  supply_fw[p,f,w,1] + inventory[p,f,1];
```

```
C_InventoryBalance {p in rPRODUCT, f in rFACTORY,  
t in 2..NT}:
```

```
  inventory[p,f,t-1] + make[p,f,t] =  
  sum { d in rDEALER : maxcost_fd > 0}
```

```
supply_fd[p,f,d,t] +  
sum {w in rWAREHOUSE: maxcost_fw > 0}  
supply_fw[p,f,w,t] +  
inventory[p,f,t];
```

Listing 6: AMPL Model File

B.2 Data File : Supply Chain Management

param prodCost :

```
    Southall Leeds :=  
Skirts  1.5  1.8  
Shirts  0    7  
Jeans   7    6.2;
```

param prodCapacity :

```
    Southall Leeds :=  
Skirts  36    54  
Shirts  0    60  
Jeans   85    36;
```

param initInv :

```
    Southall Leeds :=  
Skirts  0    5  
Shirts  5    7  
Jeans   3    5;
```

param init_invwarehouse :

```
    Liecester Norwich :=  
Skirts  0    5  
Shirts  5    7  
Jeans   3    5;
```

param invCost :


```
        Southall  Leeds :=
Skirts  0.1      0.5
Shirts  0        0.4
Jeans   0.1      0.3;
```

param invCapacity :

```
        Southall  Leeds :=
Skirts  25      30
Shirts  0        25
Jeans   60      25;
```

param warehouseCapacity :

```
        Liecester  Norwich :=
Skirts  25      30
Shirts  0        25
Jeans   9        25;
```

param inv_warehouse_Cost :

```
        Liecester  Norwich :=
Skirts  0.4      0.5
Shirts  0        0.4
Jeans   0.3      0.3;
```

param trans_fw:

```
        Liecester  Norwich :=
Southall 0.1      0.3
Leeds    0.1      0.3;
```

param trans_wd:

	London	Paris	Wien	:=
Liechester	0.5	1.1	1.3	
Norwich	0.3	0.9	1.1;	

param trans_fd:

	London	Paris	Wien	:=
Southall	1.0	2.2	2.6	
Leeds	0.6	1.8	1.2;	

param penalty :

	London	Paris	Wien	:=
Skirts	7	7	7	
Shirts	10	10	10	
Jeans	17	17	17;	

param dealerdemand

[*,London,*]:	1	2	3	4	:=
Shirts	50	55	66	76.56	
Skirts	16	17.6	21.12	24.499	
Jeans	20	55	66	76.56	

[*,Paris,*]:	1	2	3	4	:=
Shirts	10	11	13.2	15.312	
Skirts	16	17.6	21.12	24.499	
Jeans	60	66	79.2	91.872	

```
[*,Wien,*]: 1 2 3 4 :=  
Shirts      30  33  39.6 45.936  
Skirts      20  22  24.4 30.624  
Jeans       30  33  39.6 40.936;
```

Appendix C AMPL Files

C.1 Model File : Asset Liability Management

```
set ASSETS;
set rASSETS; set tAssets;
param NT;
set TP :=1..NT;

set assetclasses={'BN','EQ','CM'}ordered;
param assetType{ASSETS} symbolic;
set asset{i in assetclasses} :=
  {a in ASSETS : assetType[a] == i};
set sectors;
param assetSector{ASSETS} symbolic;
set assetBySector{s in sectors}:=
  {a in ASSETS: assetSector[a] == s};

param minreq{assetclasses}default 0;
param maxreq{assetclasses}default 0;
param ifreq{assetclasses}default 1;
param ifsectorreq{assetclasses}default 1;
param includesector{sectors}default 1;
param minsector{assetclasses,sectors}default 0;
param maxsector{assetclasses,sectors}default 100;
param g;
param tbuy := 1 + g;
param tsell := 1 - g;
```

```

param price{ASSETS, TP};
param initialholdings{ASSETS} := 0;
param cashinflow{TP} default 0;
param liabilities{TP};

var Hold{rASSETS, TP} >=0;
var Buy{rASSETS, TP} >=0;
var Sell{rASSETS, TP} >=0;
var MarketValue{assetclasses,TP} >=0;
var MarketValuetot{TP} >=0;# Total Market Value

# Objective
maximize wealth: MarketValuetot[NT];

subject to
C_mv_assetclass {t in TP,i in assetclasses :
  maxreq[i] > 0 or minreq[i] > 0 } :
  MarketValue[i,t] =
    sum{a in rASSETS : assetType[a] = i }
    Hold[a,t] * price[a,t];
C_am_total{t in TP}:
  MarketValuetot[t] =
    sum{a in rASSETS} Hold[a,t] * price[a,t];
C_sb1{a in rASSETS}:
  Hold[a,1] =
    initialholdings[a] + Buy[a,1] - Sell[a,1];
C_sb{a in rASSETS, t in 2..NT}:

```

```

Hold[a,t] =
Hold[a,t-1] + Buy[a,t] - Sell[a,t];
C_fundbalance {t in TP}:
    tsell *
    (sum {a in rASSETS} Sell[a,t] * price[a,t]) -
    liabilities[t] + cashinflow[t] =
    tbuy *
    (sum{a in rASSETS} Buy[a,t] * price[a,t]);
C_assetclass_max {t in TP,i in assetclasses :
maxreq[i] > 0 }:
    MarketValue[i,t] <=
    (1/100) * maxreq[i] * MarketValuetot[t];
C_assetclass_min {t in TP,i in assetclasses :
minreq[i] > 0 }:
    MarketValue[i,t] >=
    (1/100) * minreq[i] * MarketValuetot[t];
c_assetsectormax{t in TP,ac in assetclasses,
s in sectors:
ifsectorreq[ac] != 0 and maxsector[ac,s] < 100 }:
    sum{j in assetBySector[s]}
    Hold[j,t] * price[j,t] <=
    (1/100) * maxsector[ac,s] * MarketValue[ac,t];
c_assetsectormin{t in TP,ac in assetclasses,
s in sectors:
ifsectorreq[ac] != 0 and minsector[ac,s] > 0 }:
    sum{j in assetBySector[s]}
    Hold[j,t] * price[j,t] >=

```

```
(1/100) * minsector[ac,s] * MarketValue[ac,t];
```

C.2 Data File : Asset Liability Management

```
param NT := 4;
param g := 0.025;
param maxreq :=EQ 40;
param minreq :=BN 60;
param liabilities :=
1 0
2 1000000
3 1200000
4 1250000;
param cashinflow :=
1 150000000;
```

C.3 Script File : Asset Liability Management

This file reads the value from data file above and data tables given in appendix A with respect to the ALM problem.

```
data data\CommonData.dat;
param ConnectionStr symbolic = "MySQLDSN";
table tbl_assets "ODBC" (ConnectionStr) "asset":
  ASSETS <- [AssetName],
  assetType ~ AssetType, assetSector ~ AssetSector;
read table tbl_assets;
```

```
table tbl_prices "ODBC" (ConnectionStr)
  "asset_prices": [AssetName, TimePeriod], price~price;
read table tbl_prices;
```
