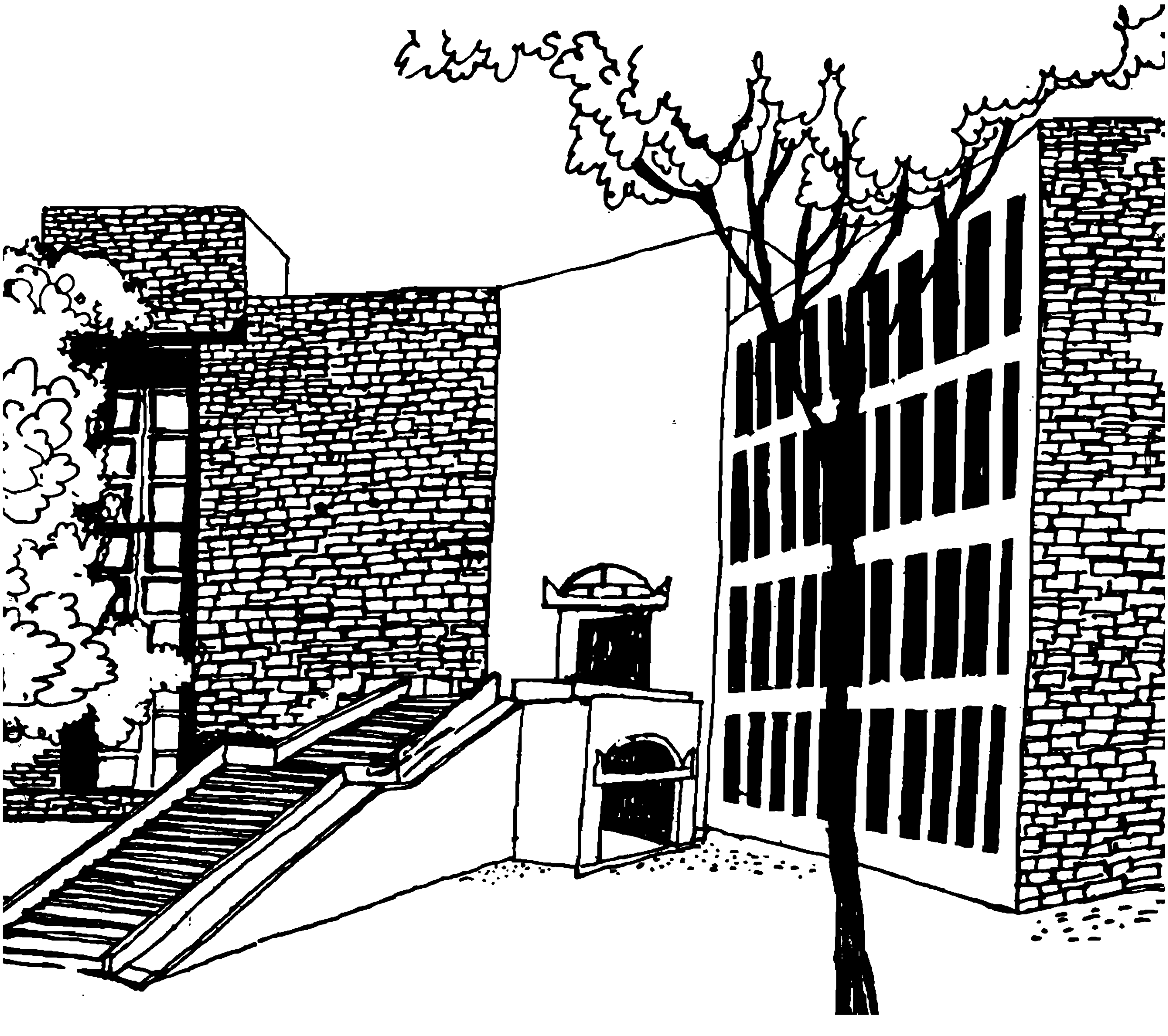
विद्याविनियोगाद्विकासः

IIM

AHMEDABAD

Working Paper



DATABASE STRUCTURE FOR A CLASS OF MULTI-PERIOD MATHEMATICAL PROGRAMMING MODELS

By

Goutam Dutta
Robert Fourer

W.P.No.2000-01-06
January 2000 1576

The main objective of the working paper series of the IIMA is to help faculty members to test out their research findings at the pre-publication stage.

INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD-380 015
INDIA

Database Structure for a Class of Multi-Period Mathematical Programming Models

BY

Goutam Dutta

Indian Institute of Management, Ahmedabad

Vastrapur, Ahmedabad 380015, India

Robert Fourer

Department of Industrial Engineering and Management Science

Northwestern University, Evanston, IL 60208, USA

 We acknowledge the financial support from American Iron and Steel Industry and an American Steel Company. This report could not be made public because of a confidentiality agreement between Northwestern University and an American Steel Company.

Abstract

We introduce how a generic multi-period optimization based decision support system (DSS) can be used for strategic and operational planning in a company with five fundamental elements, namely, Materials, Facilities, Activities, Times and Storage-Areas. This DSS which optimizes the company's activities over multiple-time horizon, having a multi-material, multi-facility, multi-activity system, requires little or no managerial knowledge of optimization techniques. We discuss the issues of interface design, data reporting and updating, production and profit planning. We also compare the performance of two different types of database structures with respect to optimization.

1. Introduction

This work started as a project to design a decision support system (DSS) for strategic planning for steel companies of North America. As the project was supported by AISI (American Iron and Steel Institute), the DSS was a generic one. An earlier publication (Fourer, 1997) discusses various aspects of database structures for mathematical programming models. While this work discusses the derivation of fundamental principles of database construction for the specific case of large-scale mathematical programming, it was only a single period model. Major steel makers in the USA could not use this generic DSS as it could not be applied to dynamic or multi-period cases. In this paper, we extend the work of Fourer (1997) in multi-period case. We introduce how a generic multi-period optimization based decision support system (DSS) can be used for strategic and operational planning in a company with five fundamental elements, namely, Materials, Facilities, Activities, Times and Storage-Areas.

In this paper we also discuss the following points in implementation:

1. What are difficulties of implementation of a multi-period DSS?
2. What are the features of multi-period DSS?
3. What are ways the indexing can be done in a multi-period database?
4. How many ways the optimal result can be represented in a multi-period DSS?
5. Why an update mode is difficult in a multi-period database?
6. What are the different possible combinations of data-structures for the multi-period mathematical programming model and how does the performance of different combinations vary with respect to data storage, data retrieval and optimization.

In section 2 of this paper, we discuss the design issues related to the database. We introduce the various elements of the DSS in this section and then discuss the implementation of the DSS. We also discuss the correspondences of the various files in the DSS and the various variables in the linear program. In section 3, we discuss the various steps of optimization: constraints generation, variable generation, matrix generation (writing of an LP file), solving of an LP file and reading of the optimal values by the database. We also discuss how we take care of the soft capacities with the help of an artificial variable. In section 4, we discuss the various features of the DSS and how the DSS can be useful for the strategic and operational planning of a process industry. Section 5 discusses the various features for reporting and updating of the data. In section 6, we compare two different variation of the database design (one with primarily hierarchical and another with primarily relational concept) with respect to optimization.

2. Database Design Issues for Multi-Period Models

The formulation of the mathematical programming model is discussed in the Appendix. In this section we discuss the various implementation issues. The single period formulation is discussed in a previous publication (Fourer, 1997).

2.1 Definitions

In this subsection, we define five fundamental elements of the model:

Times: These are the periods of the planning horizon, represented by discrete numbers (1, 2, 3 . . .).

Materials: Any product in the steel company in any stage of production (input, intermediate, output) is considered to be a Material. Materials are indexed by subscript in the formulation.

Facilities: A facility is a collection of machines that produces some materials from others. For example, a Hot Mill that produces sheets from slabs is a facility.

Activities: At any time, each facility houses one or more activities, which use and produce material in certain proportions. In each activity at each time, we have one or more input materials being transformed to various output materials. Production of hot metal, production of billets, pickling, and galvanizing are examples as activities.

Storage-Areas: These are the warehouses where raw materials, intermediate products, and finished products are stored.

2.2 Implementation of DSS

The formulations described in the Appendix-1 are implemented in 4th Dimension, a database management system and development environment. The name of the structure of the database is STEEL-TIME, which is shown in Figure 1. Files of *Materials, Facilities* and *Activities, Storage-Areas* and *Times* are also indicated. Because of inherent advantages of hierarchical database in ease of use, data storage and data retrieval, we retain the hierarchical structure for Materials and Facilities; however, other parts of the structure of the database are relational. The [Times] file is related to various time dependent parameters in the [Materials], [Facilities], [Activities], and [Storage-Areas] files.

The [Constraints] and [Variables] files are similar to those of the single period model reported in Fourer (1997). However, we have an additional field to indicate the time dimension of each variable and each constraint. Moreover, we have the additional fields of [Constraints]Dual in the [Constraints] file and [Variables]Optimal in the [Variables] file. These fields are helpful for our implementation, and are essential to conduct some experiments as discussed in section 6.

In section 6, we will discuss two different variations of this database called STEEL-TIME1 and STEEL-TIME2. The single period model described by Fourer (1997) and its database structure will be referred to as STEEL (Figure 2).

Data Management

The first decision we had to make was the choice of the database software with which our system would be implemented. As per the single period model we decided to retain the 4th Dimension relational database management software running on Apple Macintosh. Its selection was mainly based on its powerful interface features and appealing menus. Another important point is its portability. At present, 4th Dimension (version 3.5) is also available for Windows 95 and Windows 3.1. In contrast to the first version of 4th Dimension that supported purely hierarchical structures, the later versions have more relational structures.

Another important factor in designing the data structure for the multi-period model is identifying the parameters which are functions of time and making a separate subfile for time dependent variables and parameters. We will briefly discuss the implementation of each file where the data is stored.

Time File

The [Times] file (Figure 3) which has [Time]TimeID and [Time]TimeName fields. TimeName has the time identification field, the name of the time period. The user will decide whether he or she would like to plan daily, monthly, quarterly or yearly and indicate the name of the month or quarter in the field [Time]TimeName accordingly. This is the one file which is related to different time-dependent fields or subfileds of [Materials], [Facilities], [Activities] or [Storage-Areas] file.

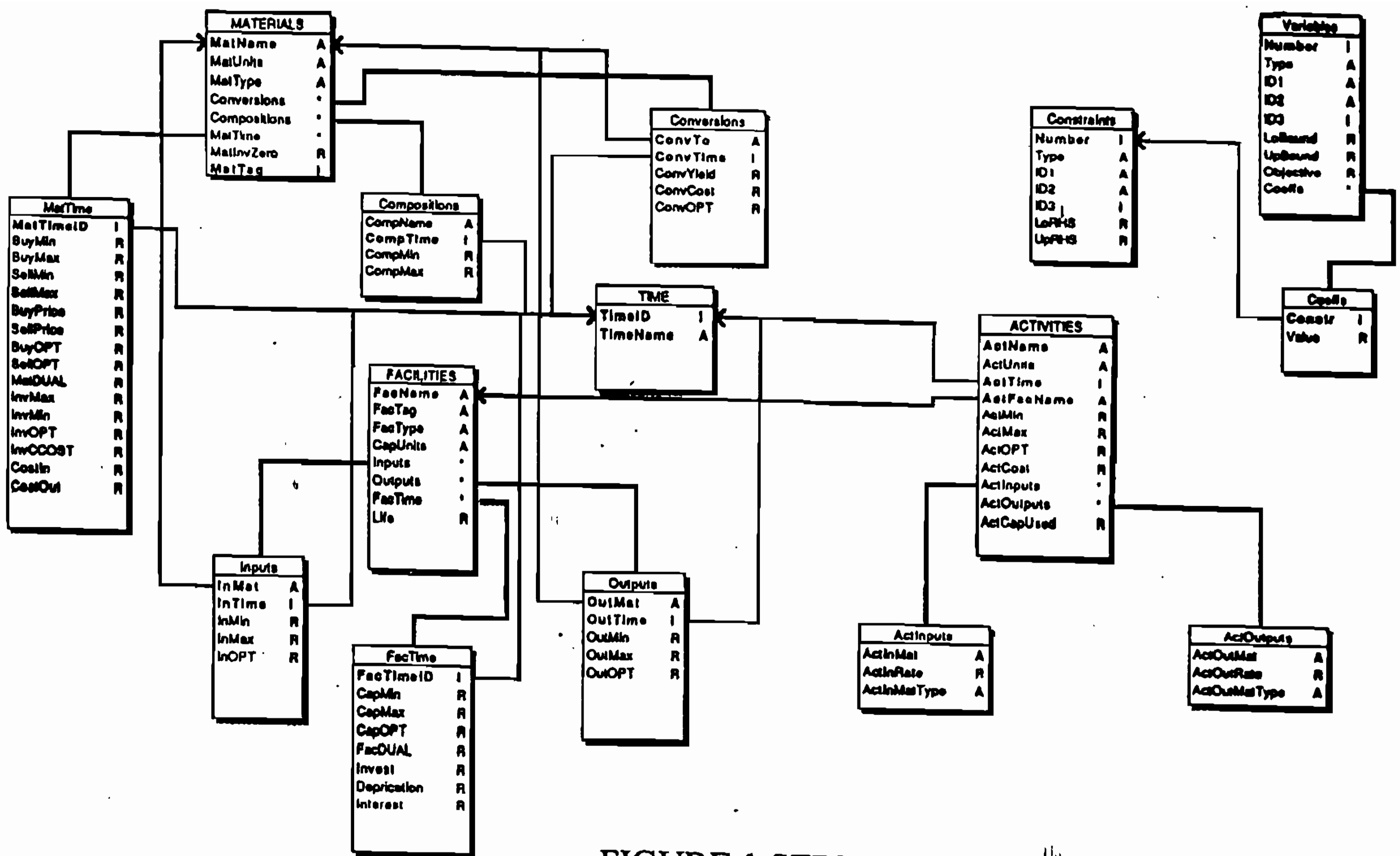


FIGURE 1 STRUCTURE OF STEEL-TIME

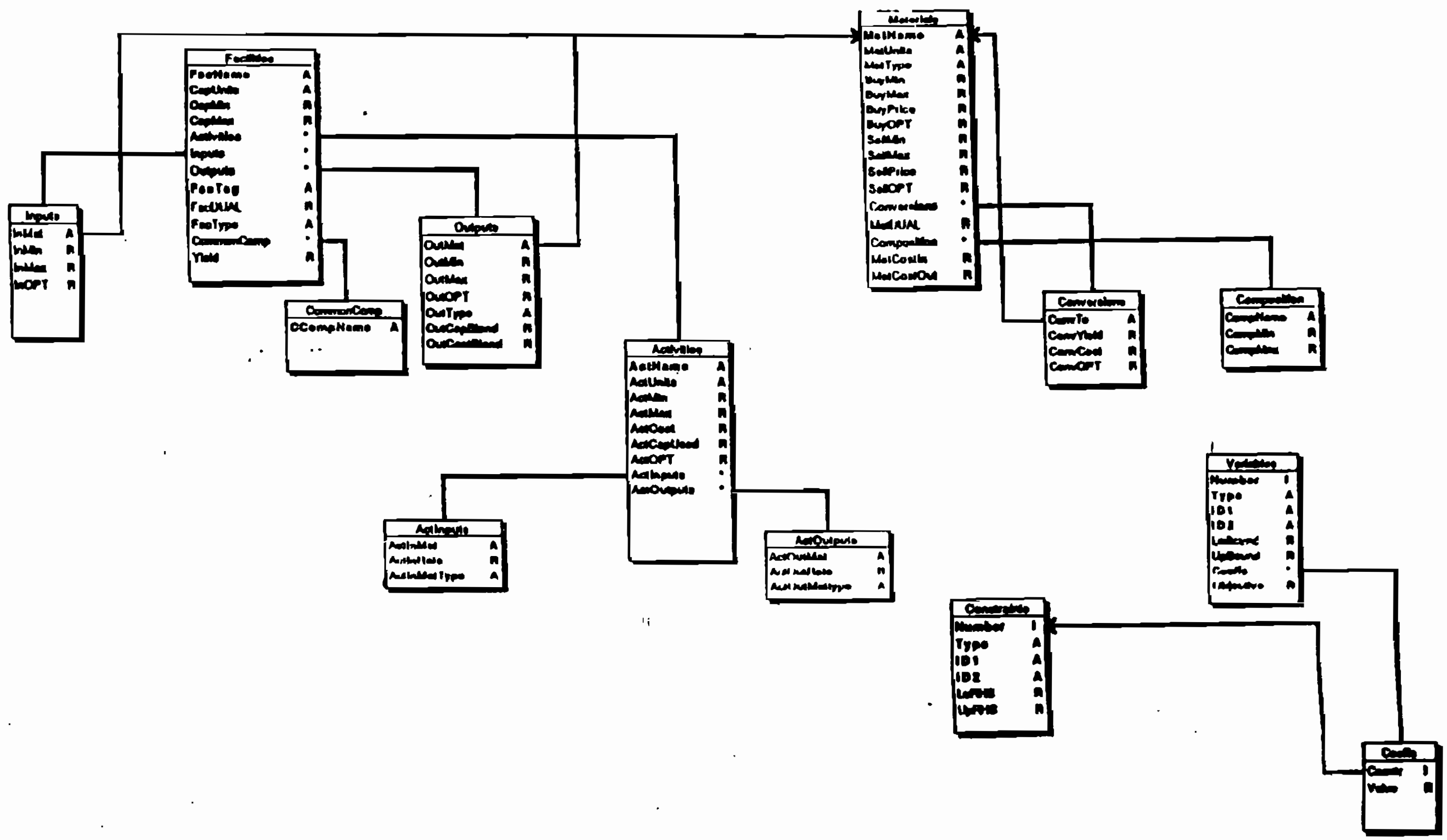


FIGURE 2 STRUCTURE OF STEEL

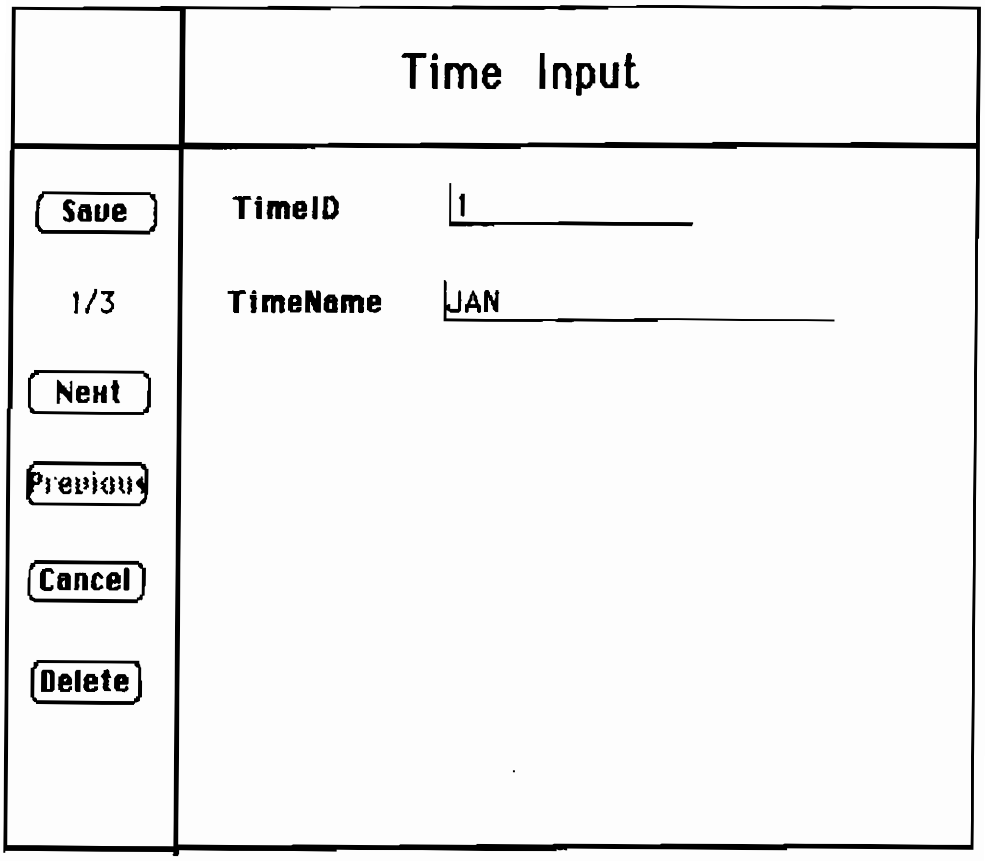


FIGURE 3 TIME INPUT FILE

Materials File

The Materials data (Figures 4 and 5) is stored in a hierarchical way. In Table 1, we show the one to one correspondence between the parameters of the LP model and the fields on the[Materials] file. In this file the material name([Materials]MatName) and material identification string ([Materials]MatTag) are unique. [Materials]MatTag is required for data entry in the subfiles of the [Materials] file as discussed in Chapter 8 in detail. In the [Materials] file, BuyMax, BuyMin, SellMax, SellMin, BuyPrice, SellPrice, BuyOpt, SellOpt, InvMax, InvMin, InvOpt, InvCCost, CostIn and CostOut are the time-dependent subfields in the [Materials]MatTime subfile. Since the model is multi-period, the dual variables (MatDual) are also considered as a function of time and are put in the MatTime subfile. MatName, MatUnits and MatType are the main fields of the file. MatTimeID is the indexed subfield of the Materials[MatTime] subfile. For each material, there is a record of the [Materials]MatTime subfile corresponding to each record of the Time file; the data in each [Material]MatTime'MattimeID subfile field is the same as the value in the corresponding [Time]TimeID field.

The subfile Conversions is the second subfile of [Materials] file. This subfile is indexed by two subfields : Conversion time (ConvTime) and Conversion Material (ConvTo). In addition, it has conversion cost (ConvCost) and conversion yield (ConvYield) as additional subfields. The [Materials]Conversions subfile is similar to the analogous subfile in the single period model in STEEL (Figure 2) except that it has the additional subfield [Materials]Conversions'ConvTime and is indexed over times as well as materials.

The Materials file has a third subfile called [Materials]Compositions. In this subfile, we have [Materials]Compositions.'CompName and [Materials]Compositions'CompTime. [Materials]Compositions'CompName is the time-dependent subfield of the subfile. The maximum and minimum composition of each element or compound are the two additional subfields. These subfields are required for the *Cost Allocation Model* that we do not discuss in this paper.

Table 1

Correspondence of [Materials]File and the LP Model

l^{buy}_{jt}	[Materials]MatTime'BuyMin
u^{buy}_{jt}	[Materials]MatTime'BuyMax
c^{buy}_{jt}	[Materials]MatTime'BuyPrice
l^{sell}_{jt}	[Materials]MatTime'SellMin
u^{sell}_{jt}	[Materials]MatTime'SellMax
c^{sell}_{jt}	[Materials]MatTime'SellPrice
l^{inv}_{jt}	[Materials]MatTime'InvMin
u^{inv}_{jt}	[Materials]MatTime'InvMax
h_{jt}	[Materials]Mattime'MatInvCCOST
x^{inv}_{j0}	[Materials]MatInvZero
$a^{conv}_{jj't}$	[Materials]Conversion'ConvYield
$c^{conv}_{jj't}$	[Materials]Conversion'ConvCost
$Comp^{\max}_{j\alpha t}$	[Materials]Compositions'CompMax
$Comp^{\min}_{j\alpha t}$	[Materials]Compositions' CompMin

Custom

MAT-TAG	MATERIAL	UNITS	MATERIAL TYPE
101	CC BILLET	TONS	Intermediate
102	SINTER	TONS	Input
105	SLAG	TONS	Intermediate
106	HOT METAL	TONS	Intermediate
110	ORE	TONS	Input
201	COAL	TONS	Input
202	COKE	TONS	Intermediate
213	NEW MAT	TONS	Input
401	CRUDE STEEL	TONS	Input
402	STEEL for CC	TONS	Intermediate
501	HEAVY MELTING SCRAP	TONS	Intermediate
503	STEEL SCRAP	TONS	Intermediate
503	LIGHT MELTING SCRAP	TONS	Intermediate
504	MILL SCRAP	TONS	Intermediate
601	BILLET	TONS	Intermediate
602	BLOOM	TONS	Intermediate
603	SLABS	TONS	Intermediate

Add Entry

Done

FIGURE 4 OUTPUT LAYOUT OF MATERIALS FILE

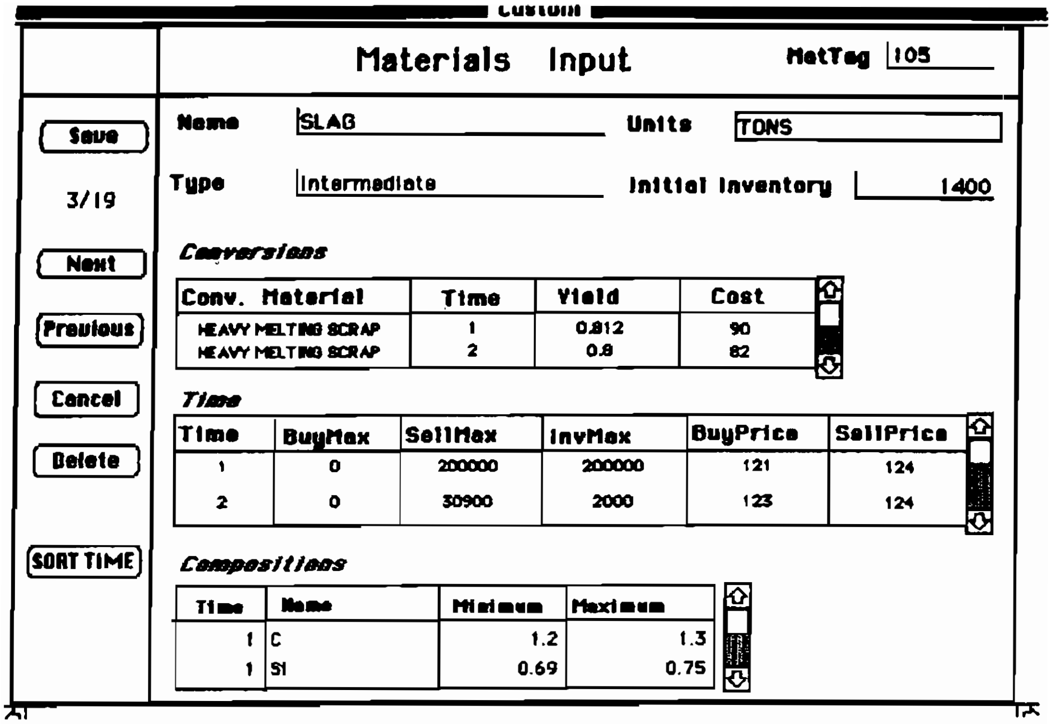


FIGURE 5 INPUT LAYOUT OF MATERIALS FILE

Facilities File

In the Facilities file (Figures 6 and 7), for time dependent parameters we retain structure similar to that of the [Materials] file. In Table 4.2, we show the one to one correspondence between the parameters of the LP model and the fields [Facilities] file. We define[Facilities]FacTime as a subfile where the CapMax, CapMin, CapOPT and CapDual subfields are the time dependent maximum, minimum, and optimal production level of the facility, and the time dependent dual value of the facility capacity. The VendorCost is the cost of vendoring (outsourcing) an additional unit capacity of the facility at that time.

There are two indexed subfields in [Facility]Inputs, which is a subfile of the Facilities file. The first one is the input material, which is related to the [Materials] file. The other is [Facility]Inputs'InTime which is the time dependent field of the [Facilities]Input File and is related to the Time file. The subfile [Facilities]Outputs subfile is entirely analogous.

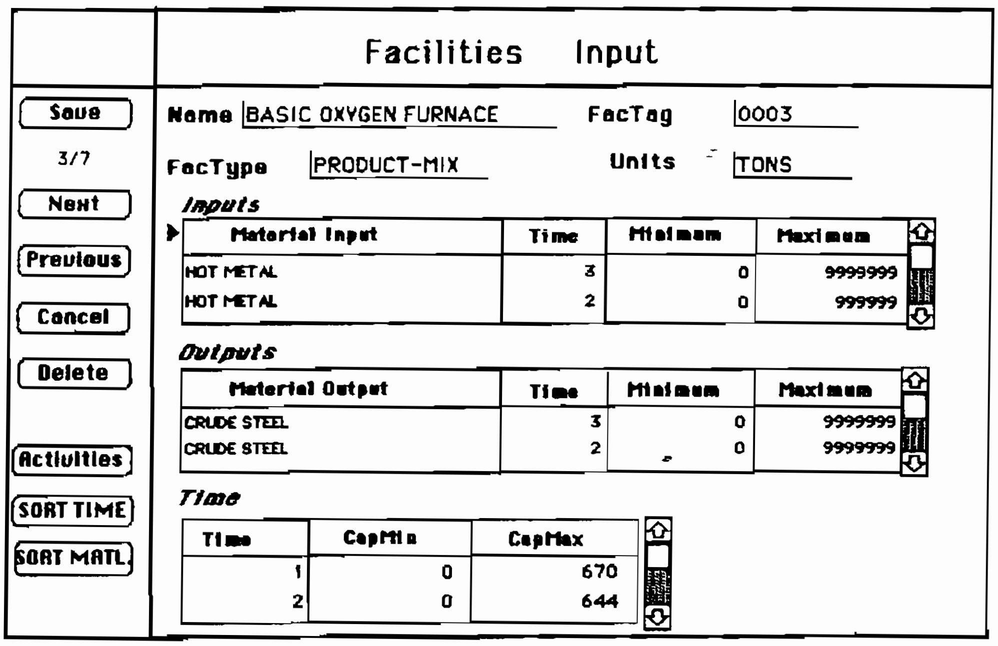


FIGURE 6 INPUT LAYOUT OF FACILITIES FILE

FACILITY NAME	FAC TAG	FAC TYPE	CAPACITY UNITS
BLAST. FURNACE	0001	PRODUCT-MIX	TONS
COKE OVENS	0002	PRODUCT-MIX	TONS
BASIC OXYGEN FURNACE	0003	PRODUCT-MIX	TONS
CONTINUOUS CASTER	0004	PRODUCT-MIX	TONS
ROLLING MILL NO 1	0005	PRODUCT-MIX	HOURS
MERCHANT MILL NO. 1	0006	PRODUCT-MIX	TONS
S.B.B. MILL 1	0007	PRODUCT-MIX	TONS

Add Entry

Done

FIGURE 7 OUTPUT LAYOUT OF FACILITIES FILE

Table 2

Correspondence of [Facilities] File and the LP Model

l_{ijt}^{in}	[Facilities]Inputs'InMin
u_{ijt}^{in}	[Facilities]Inputs'InMax
l_{ijt}^{out}	[Facilities]Outputs'OutMin
u_{ijt}^{out}	[Facilities]Outputs'OutMax
C_{it}^{vend}	[Facilities]FacTime'Vendor_Cost
l_{it}^{cap}	[Facilities]FacTime'CapMin
u_{it}^{cap}	[Facilities]FacTime'CapMax

Activities File

[Activities] is defined (Figure 8 and 9) as a separate file. (In STEEL-TIME2, we consider [Activities] as a subfile of the [Facilities] File.) There is a field of [Activities]ActTime which is the indexed field of time in the [Activities] file and related to the [Times] file. In Table 4.4, we show one to one correspondence between the parameters of the LP model and the [Activies] file. In each activity there is a field, ActFacNamethat specifies which facility it belongs to. This is required so that the user can search for the activity through the facility. The [Activities] file can be indexed over [Activities]Act Name or [Activities]ActTag (identification string). Two activities may have the same ActName (like PRODUCTION OF BILLET), but if they have a different ActTime, they will have different a ActTag. In other words every record of [Activities] file will be identified by an unique [Activities]ActTag.

While defining the activity inputs (ActInMat) or activity outputs (ActOutMat), we have to consider the fact that ActInMat (or ActOutMat) should have only those materials which are in Facility Inputs (or Outputs) and also at the time where ActTime is equal to [Facilities]Inputs'IntTime ([Facilities]Outputs'OutTime). The other important field is ActTag, the unique identification. For example, let us assume that the BLOOM, BILLET and SLAB are available as [Facilities]Inputs at Time =1 in [Facilities]FacName =ROLLING MILL, but BLOOM and SLAB are only available as [Facilities]Inputs at Time =2 in the same facility. In the [Activities]File at Time=1 the possible choices available in the subfield Activities]ActInPuts'ActInmat are BLOOM, BILLET and SLAB, but only SLABand BLOOM are available as [Activities]ActInputs'ActInMat at Time =2 in the same facility.

Table 3

Correspondence Of [Activities] File And The LP Model

l^{act}_{ikt}	[Activities]ActMin
u^{act}_{ikt}	[Activities]ActMax
u^{act}_{ikt}	[Activities]ActCost
r^{act}_{ikt}	[Activities]ActCapUsed
a^{in}_{ijkt}	[Activities]ActInputs'ActInMat
a^{out}_{ijkt}	[Activities]ActOutPuts'ActOutMat

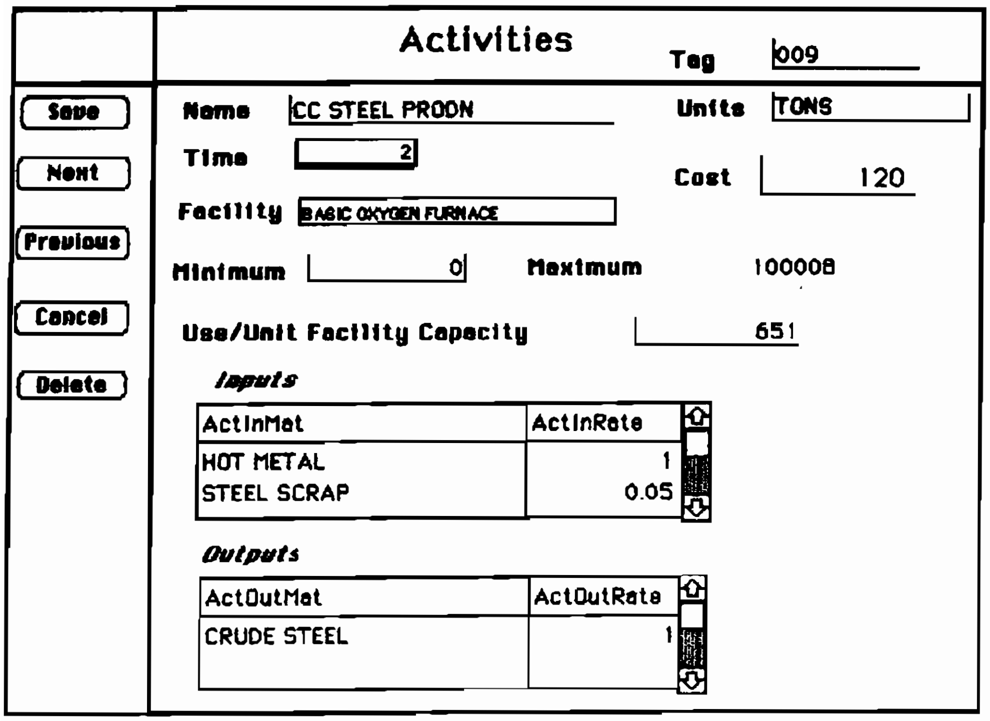


FIGURE 8 INPUT LAYOUT OF ACTIVITIES FILE

Activity Name	Time	Minimum	Maximum	Cost	ActTag
BILLET PRODN AT SBBM	3	0	67000	120	024
BILLET PRODN AT SBBM	2	0	95000	67	023
BILLET PRODN AT SBBM	1	0	40000	45	022
WIRE RODS PRODN	3	0	30000	45	021
WIRE RODS PRODN	2	0	30000	34	020
WIRE RODS PRODN	1	0	30000	23	019
PRODN OF BLOOM	3	0	98000	23	018
PRODN OF BLOOM	2	0	92000	18	017
PRODN OF BLOOM	1	0	100000	43	016
CC BILLET PRODN	3	0	50000	123	015
CC BILLET PRODN	2	0	60000	89	014
CC BILLET PRODN	1	0	50000	67	013
CRUDE STEEL PRODN	3	0	9999999	123	012
CC STEEL PRODN	3	0	100000	11	011
CRUDE STEEL PRODN	2	0	9999999	123	010
CC STEEL PRODN	2	0	100008	120	009
CRUDE STEEL PRODN	1	0	9999999	12	008
CC STEEL PRODN	1	0	120000	10	007
PRODN OF COKE	3	0	9999999	120	006
PRODN OF COKE	2	0	9999999	56	005
PRODN OF COKE	1	0	9999999	45	004

FIGURE 9 OUTPUT LAYOUT OF ACTIVITIES FILE

Storage-Areas File

In the [Storage-Areas] file (Figures 10 and 11) we have the name of the Storage-Area and the time at which the materials are stored. In addition, we have the capacity constraint of the storages giving the maximum and the minimum capacities of the storage-areas. The structure of the [Storage-Areas] file is similar to that of the [Activities]File. [Storage_Areas]StoreTag is the field which uniquely identifies the records of the file. In the[Storage-Areas] file, we have a subfile called [Storage-Areas]StoreMatList which lists all the materials that can be listed.

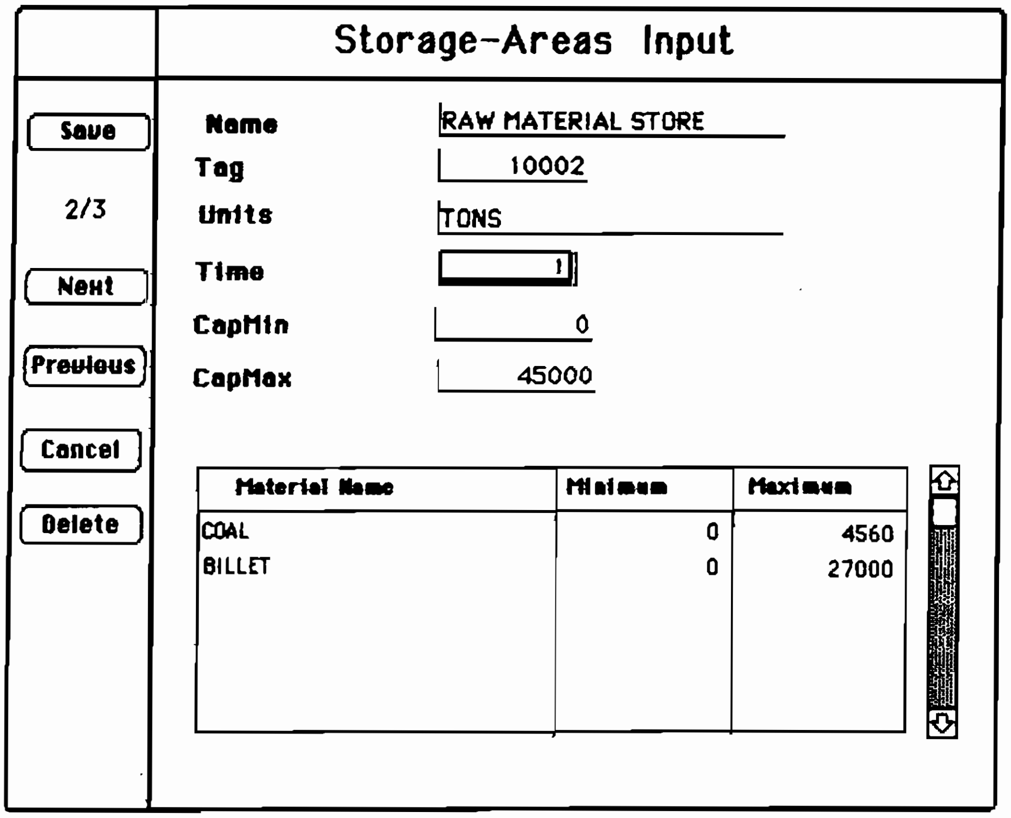


FIGURE 10 INPUT LAYOUT OF STORAGE-AREAS FILE

Table 4

Correspondence Of [Storages-Areas] File and the LP Model

l_{st}	[Storage-Areas]StoreMin
u_{st}	[Storage-Areas]StoreMax

Custom

StorageName	StoreTag	StoreTime	StoreUnits	CapMin	CapMax
FINISHED GOOD STORE	10001	1	TONS	0	27000
RAW MATERIAL STORE	10002	1	TONS	0	45000
STORE AT OPEN SPACE	10004	3	TONS	0	80000

Add Entry   Done

FIGURE 11 0UTPUT LAYOUT OF STORAGE-AREAS FILE

Variables File

In the [Variables] file (Figure 12) we have fields Number,Type (Material Bought, Material Sold, Material Inventoried, Activity at Facility), Identification Number 1 (ID1), Identification No 2 (ID2), Objective, Upper bound and Lower bound as in the single period model. However, we have also an Identification Number 3 (ID3) field which indicates the time of the variable. [Variables]Optimal refers to the most recent optimal value of the variable. The variables file has a subfile known as [Variables]Coeffs which has a subfield called [Variables]Coeffs'Constr and this constraint is related to the [Constraints]Number of the [Constraints] file.

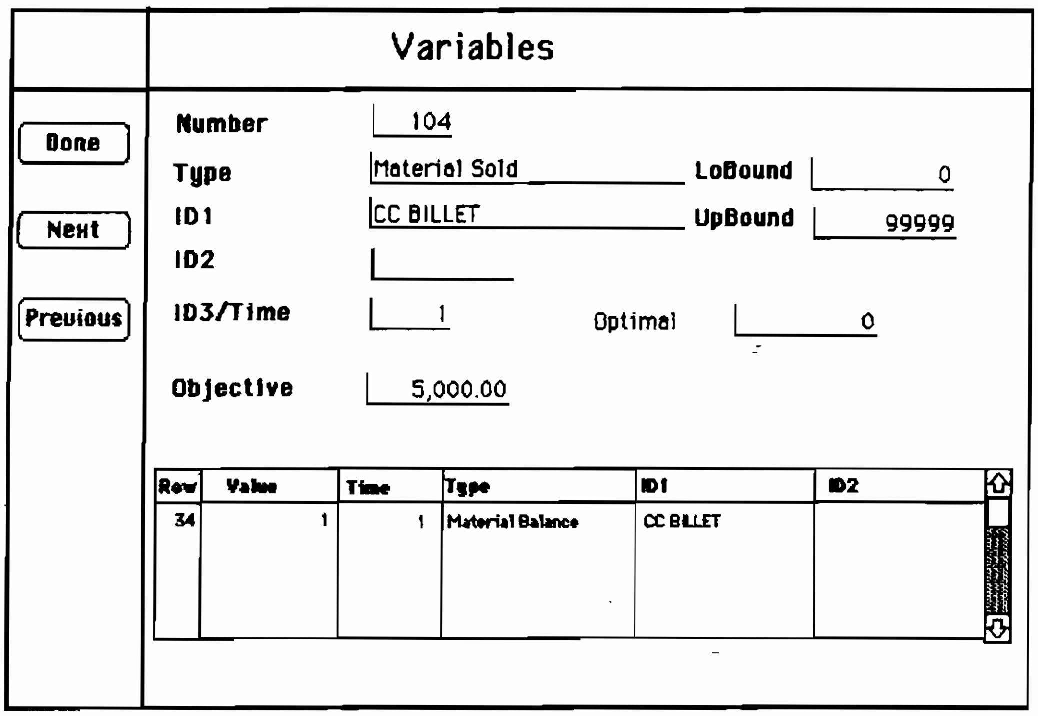


FIGURE 12 INPUT LAYOUT OF VARIABLES FILE

Constraints File

In addition to [Constraints]Number, the [Constraints] file (Figure No.13) has a field for Type (Material Balance, Facility Input, Facility Output and Facility Capacity, Storage Capacity or Storage Total, which They refer to the equation numbers 3-8 respectively in Appendix). The Identification Number 1 ([Constraints]ID1) indicates the Material Name for the Material Balance equation and Facility Name for the other three types of constraints. The Identification Number 2 ([Constraints ID2) refers to the material for the Facility Input and Output respectively. As in the [Variables] file, ID3 refers to the time of the variable. [Constraints]Dual refers to the dual variable corresponding to the most recent optimal solution.

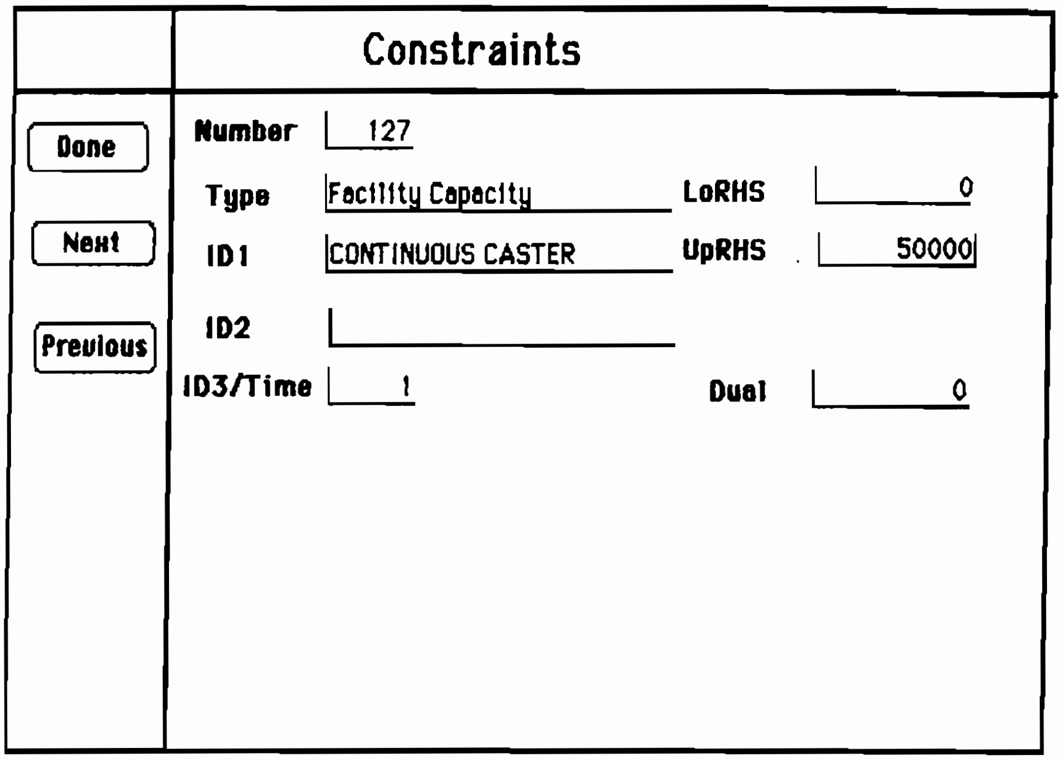


FIGURE 13 LAYOUT OF CONSTRAINTS FILE

3. Optimization Steps

Once the data of the five files and their respective subfiles are entered, they are validated by a set of diagnostics which will be explained in section 4.4. This subsection describes how the optimization routine is performed.

The principal steps (Figure 14) are:

1. The data describing the production scenario at different time periods are collected and stored in the database as explained above.

2. The constraints associated with the linear program are generated. The constraint related data values, LoRHS (lowest value of Right Hand Side) and HiRHS (highest value of Right Hand Side), are extracted from the database and stored in the [Constraints] file.

3. The variables of the associated linear program are determined. The variables related to data values are stored in a separate [Variables] file. This step involves writing the non-zero coefficient in the [Variables]'Coeff subfile. In this step, the user is given a choice of discounted or undiscounted optimization. If the user is interested in discounted optimization, he or she has to enter the interest rate (Figure 15) and all the cost, price, and revenue data are then converted to their discounted values in the objective function of the model.

4. The [Constraints] and [Variables] files are scanned and all of the essential information about the linear program is written to an ordinary textfile in a compact format. This textfile is the input file to our solver.

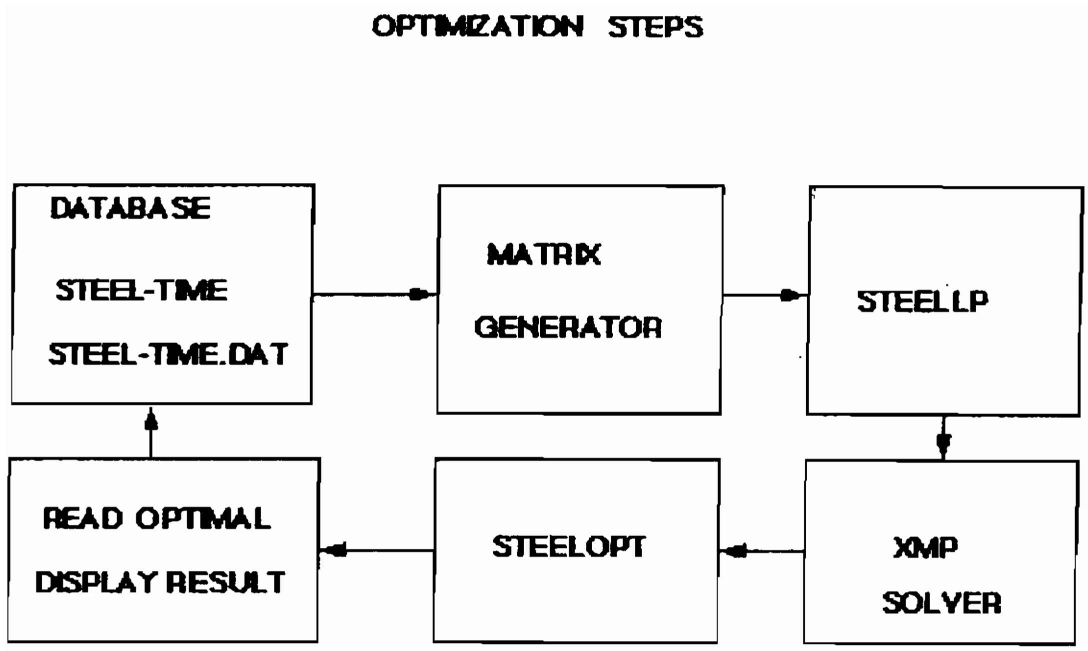


FIGURE 14 OPTIMIZATION STEPS

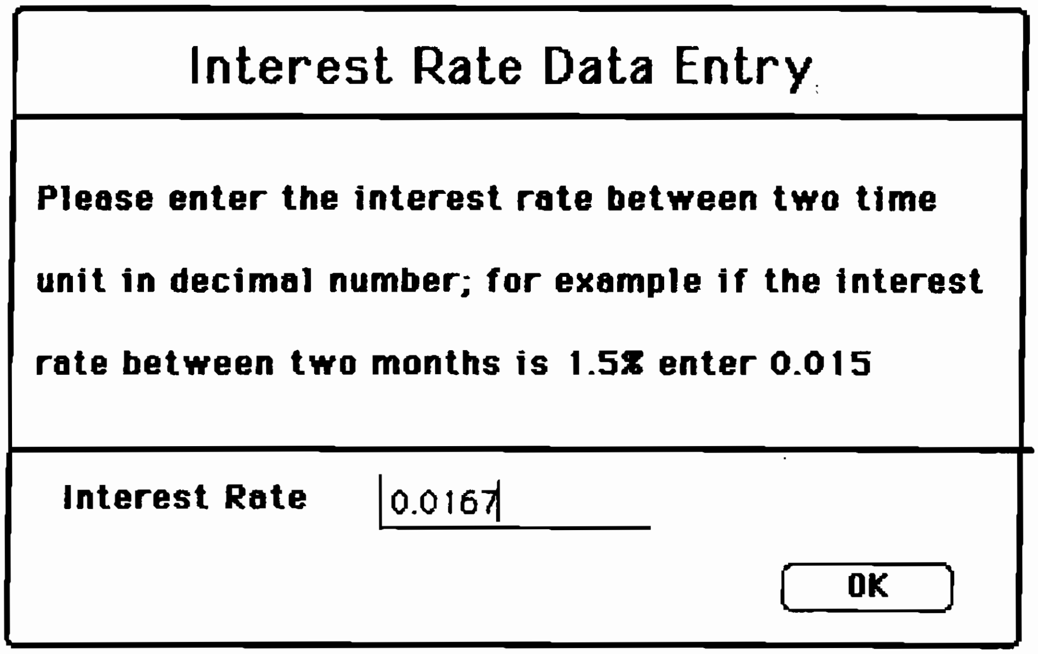


FIGURE 15 INTEREST RATE DATA ENTRY

5. The text file (steellp) is read by the XMP solver (Martsen, 1981), which solves the indicated linear program , and then writes the optimal values of the variables to a second text file.

6. The second text file is read and the optimal values are placed in appropriate fields of the [Materials], [Facilities], [Activities], and [Storage-Areas] files.

3.1 Soft Capacities

If we have an infeasibilty in the "Facility Capacity" constraint, we can generate a "Soft Capacity" variable, which is similar to an artificial variable. At the end of step 2, the user will have an option to use a procedure, which generates this variable.

This procedure will generate $X_{i,t}^{vend}$ (the soft capacity variable) in the Facility Capacity (Constraint 6 of Appendix) and will also generate its related objective function coefficients. The user needs to enter the value of [Facilities]FacTime'VendorCost which is the coefficient of the soft capacity variable in the objective functions. In case we do not want the capacity constraint to be violated, we assign a very high value to these objective function coefficients.

3.2 Diagnostics

The diagnostic routines are written to ensure that the linear program is complete and free from many errors and infeasibilities. We use the various 4th Dimension file procedures, layout procedures and global procedures to implement these routines.

The following generic diagnostics are applied to all files and subfiles or variables or constraints, as appropriate:

Rule 1. For every variable the upper bound should not be less than the lower bound. For every constraint the lower right hand side (LoRHS) should not be more than the higher right hand side (HiRHS).

Rule 2. For every variable and every constraint, there should not be more than one non-zero element.

Rule 3. For every subfile indexed over one time subfield, the number of subrecords in the subfile should be same as the number of records in the [Times] file.

Rule 4. For files and subfiles indexed over one time field and one non time field, the number of records (or subrecords) should not be more than the product of the number of records (or subrecords) in the [Times] file and the number of records related to the non-time field.

Rule 5. If a record or sub-record is indexed over a time field or sub-field and one non-time field or subfield, there will be only one record or subrecord containing any particular combination of the time field and non-time field.

We assume that the linear program is complete with respect to all time period data. If we do not have data for any period, a default value is taken. The default values of all minimums are zero and of all maximums are infinity (implemented as 99999999). The default value of yield is 100 % and of rolling rate is 1 tons/ hour.

4. Features of the DSS

We are interested in using this DSS for strategic and operational planning. We will discuss various features of this DSS in this subsection.

4.1 Strategic and Operational Planing

In strategic planning, the DSS will be able to answer questions such as:

1. What is the effect of cost or price changes of raw materials and finished products on the product-mix?

2. If we invest 20 million dollars to install a coal injection system in the blast furnace this year, anticipating an increase of productivity of the blast furnace by 5 percent in subsequent years, is the investment justified?

3. If the company is planning to diversify to different products, what products should be chosen?

In operational planning the DSS will be able to help the steel company officials with questions like these:

1. How does product-mix planning for the current month affect planning in the subsequent months, and can this monthly plan be divided into four weekly plans or even daily plans for 30 days?

2. In response to a shortage of liquid steel, which results in the partial operation of the finishing mills in the downstream production line, which of the finishing mills should go down?

3. Should external scrap be purchased as a substitute for hot metal and at what price?

For example, in the experience with the Indian steel company (Sinha et. al., 1995, Dutta et. al, 1994) the marginal profit of an extra megawatt of electrical power was found to be several million dollars. This study justified the investment of installing diesel-generating sets. Similar studies can be done using our DSS also.

4.2 User Friendliness

This is the most important point of this research. We have been able to demonstrate that multi-period, multi-product, multi-facility process industry planning can be done with little or no knowledge of linear programming. All the user has to do is click the appropriate buttons to run the linear programs.

The DSS can be used in three modes: *Data, Optimal* and *Update*. In the *Data* mode, the user enters data in the five different files. The Optimal mode is for display of optimal values and dual prices. The DSS takes a much longer time (92 minutes) to generate the [Variables] file and the [Constraints] file than to solve the problem (3 minutes). If there is no addition or deletion of records in the [Materials], [Facilities] and [Activities] file, any change in the parameters of these files can be reflected in corresponding changes to the [Variables] and [Constraints] file (without procedures of variable and constraint generation). This is accomplished in the *Update* mode resulting in saving of user time.

As a user-friendly tool for strategic planners, the dual prices for "Facility Capacity" constraints for each facility are displayed to indicate the profit improvement potential. The details of the dual prices are explained in the Chapter 5.

4.3 Multi-Period Model

The multi-period structure of our DSS has the following advantages:

1. The model can show how the cash flow of the company changes with different interest rates. The user is allowed to enter the interest rate. The user also has the option to optimize over nominal or discounted financial parameters.

2. The importance of inventories is considered in this model. Using this DSS we will be able to make decisions like whether it is more profitable to produce at the current time period and hold inventory or to produce in the future.

3. The user can see the effect of changing the parameters in one time period on the optimal decisions for other time periods.

4.4 Generality and Flexibility

The model is sufficiently generic so that it can be used by any process industry that transforms materials in different facilities. When the company decides to make any new product, a record can be added to their materials database. Similarly when a new facility is installed the user can enter an appropriate record. For any linear programming model done in AMPL or GAMS, the user does not have the advantage of route flexibility. In this DSS, any route of the product can be added or deleted by addition and deletion of appropriate material, facility and activity. If another industry wants to use this software, they only need to change the relevant data entry files for their company.

5. Reporting and Updating the Data

In this section, we consider the different files and discuss the time dependent layouts where the time dependent parameters are entered as subfields.

5.1 Layouts with Time as a Subfield

First, let us consider the [Materials] File. In this file, no time dependent parameters are in the file level except for MatInvZero. This field is required to initialize the linear programming model. [Materials]MatTime is a subfile which is indexed over time, so we have designed a layout that displays all the time-dependent parameters that are in the subfields in this subfile (Figure 16). These fields will be the same in *Data* or *Optimal* layouts. In order to see the optimal value of the material COIL bought at Time = 2, the user has to select the optimal mode in the *Examine* menu of the main menu and select Materials. Then a list of Materials will be displayed. Then the user has to select the material COIL and a layout called Materials Optimal (Figure 17) will be displayed. In this layout these will be an included layout that lists the data of all time dependent parameters of the materials COIL. Once the user selects Time=2 a list of parameters is displayed in a layout for Time=2 and one of them is BuyOPT which shows the optimal value of Material bought in Time= 2. Similarly, if the user wants to get the Buyprice of material called SCRAP at Time =3, he or she has to go through steps similar to all these.

We now discuss two different types of searches. We want to compare the searching process of an activity and an input material in the same [Facilities] file. Let us assume that [Facilities]FacName= BASIC OXYGEN FURNACE. The user selects Facilities and Optimal in the *Examine* menu of main menu and gets a listing of all facilities and selects the facility = BASIC OXYGEN FURNACE.

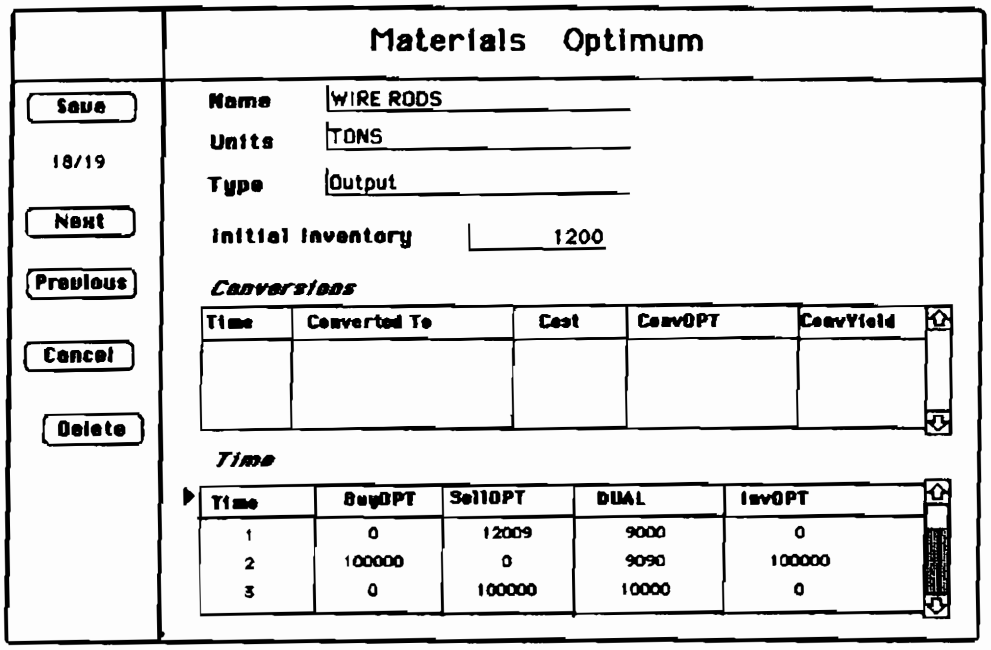


FIGURE 16 MATERIALS OPTIMAL

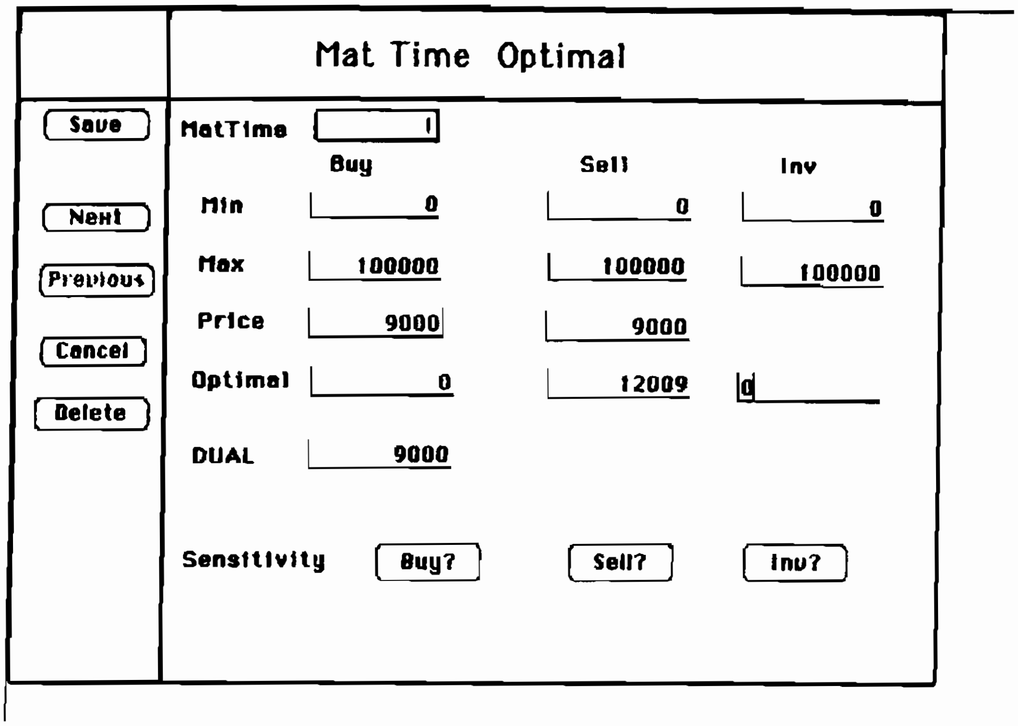


FIGURE 17 MATTIME OPTIMAL

and goes to the Facilities Optimal screen. This is common to both the searches. In the first search, he or she clicks at the Activities button and goes to the next page of the Facilities Optimal Screen. This screen layout lists all the activities in this facility as an included layout. If the user wants to find the values of rate for the output material STEEL for the Activity = CRUDE STEEL PRODUCTION at Time=2 of this facility, then he or she looks at the list of activities and searches for Activity = CRUDE STEEL PRODUCTION and Time=2. This leads to an Activity Optimal Screen which list the output materials. Then this list gives the value of output rate for the output material =STEEL. *In this case, to get a required value, we first search (on the [Activities] file) with a combination of two fields, and then look for a subfile or subfield.* In the second search, to get the maximum value of input material STEEL SCRAP that can be accommodated in this facility at Time=2, the user looks at the facilities Optimal Screen and looks at the included layout of Inputs. This included layout lists all the input materials at all times. The user then searches for Material = STEEL SCRAP and Time=2. *In this case the search is performed with two searches at the subfile level.*

5.2 Included Layouts and Graphs in the Time File

Suppose we have a question from a user. At Time =1, what is the optimal value of material sold for SINTER, and HIGH CARBON BILLET? In the Examine menu, the user can select Materials and *Optimal*, and this will lead to a list of Materials. The User can double click at SINTER and this will lead to the Materials Optimal screen of SINTER. In this screen there will be a list of Times and the user can find the optimal value of material sold at Time = 1 in this list. Then he or she has to return to the list of Materials and double click here again at HIGH CARBON BILLET. Then he or she gets another Materials Optimal Screen of HIGH CARBON BILLET. Then he or she can look again at the Time Layout and see the material bought at Time = 1. This is a cumbersome procedure. At Time=1, the user can not go from one material to another. This can be overcome by making an included layout of the [Materials] file in the [Times] file.

In the 4th Dimension database management system, we have the advantages of using an included layout. In an included layout, the layout, of one file can be included in another file. So we can see the [Materials] file or the [Facilities] file as an included layout in the [Times] File. In this case, the user Selects Time-Material at the Examine menu. This leads to a list of times. The user selects at Time=1 and he or she is supplied with a list of [Materials] at Time=1. This is displayed in Figure 18 and 19 In this case the user can switch from one material to another at the same time (Time=1).

Similar arrangements can be made for the [Facilities] file and similar advantages can be achieved out by making the [Facilities] file as an included layout of the [Times] File. This is displayed in Figures 20 and 21.

In discussing the optimal layouts, we also consider the case of graphs. We can display the graphs of the different variables, such as the materials bought, and materials sold. We have tried two different types of graphs, the line graph and the bar graph. In a similar way, we can display graphs of the material inventory (Figure 22). Other than that we can display the maximum, minimum, and the optimal values from the [Facilities] Inputs or [Facilities]Outputs subfile.

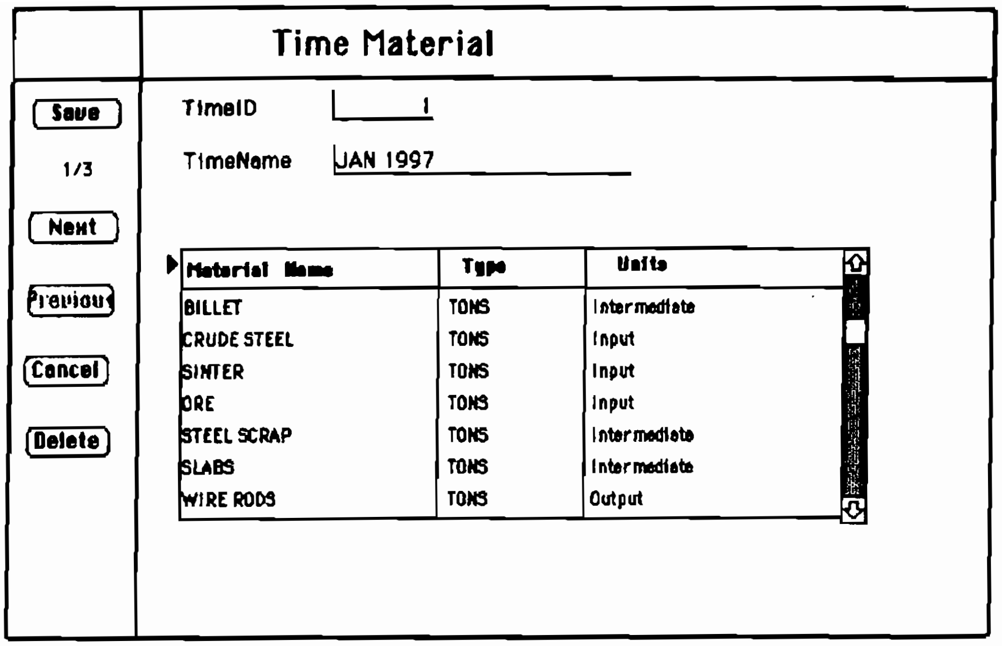


FIGURE 18 MATERIALS LIST IN TIME LAYOUT

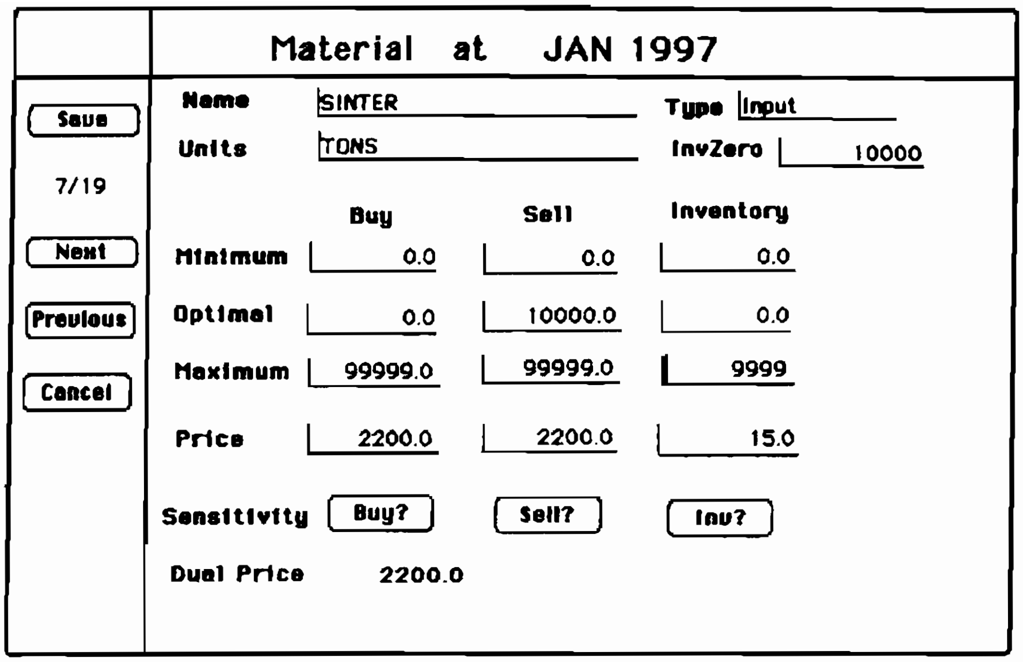


FIGURE 19 MATERIAL INPUT IN TIME LAYOUT

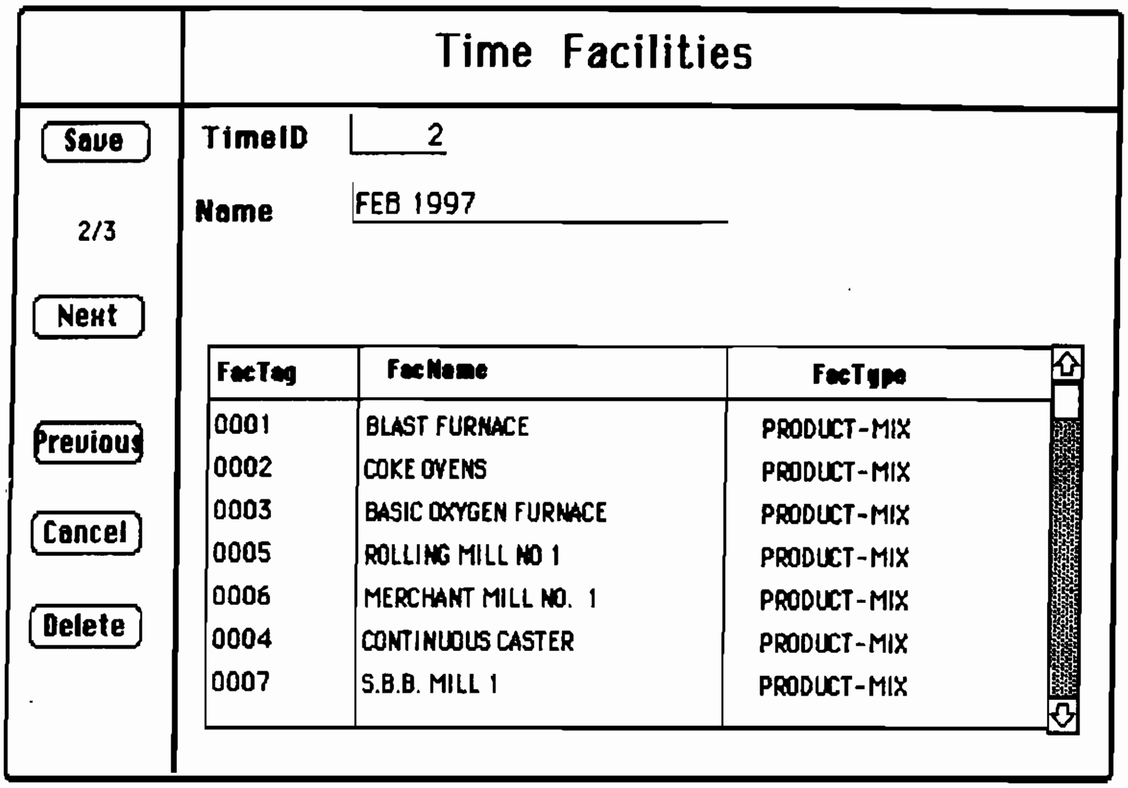


FIGURE 20 FACILITIES LIST IN TIME LAYOUT

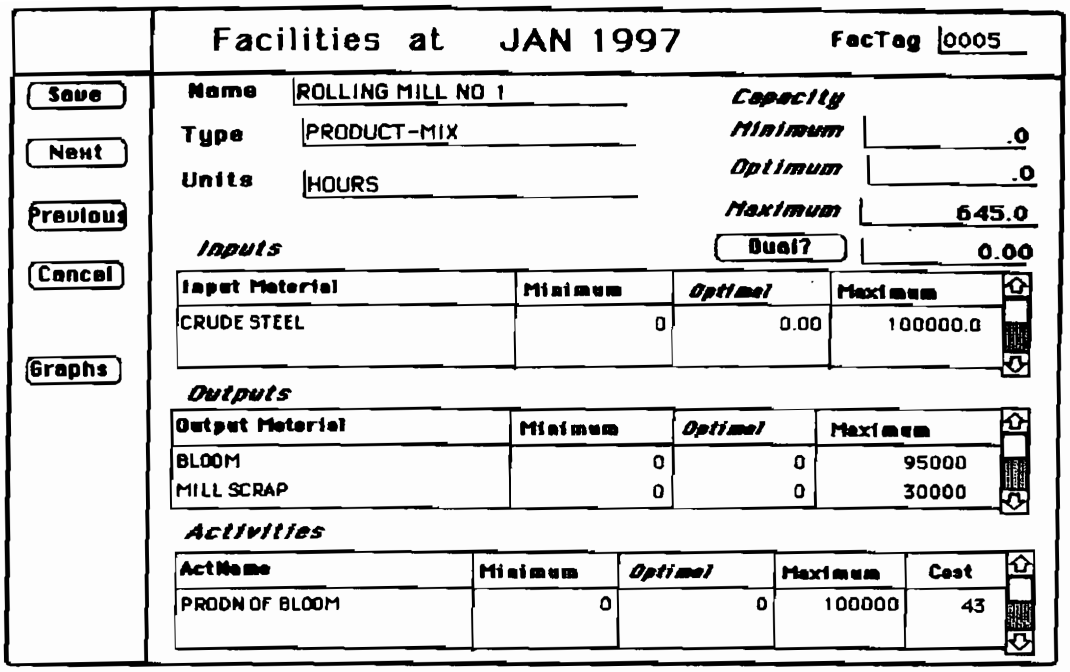


FIGURE 21 FACILITIES IN TIME LAYOUT

of the [Facilities] file. The graphs give the user an idea where the optimal value lies and how much close the optimal value is to the maximum.

5.3 Reporting of Optimal Dual Values

In this section, we discuss the difficulties in reporting the optimal dual values in the multiple time period. For a single period model, the display of dual values is simple and straightforward. However, for the multi-period model we have dual values for more than one time period. In addition to that, the reduced cost for the variable Material Inventoried any time period is dependent on dual values from more than one time period. This makes our task difficult for displaying the optimal dual values.

Let the reduced costs for the Material Bought, Material Sold and Material Inventoried at time t be denoted by RC_{jt}^{buy}, RC_{jt}^{sell}, RC_{jt}^{inv} respectively, and let Π_{jt} be the dual price of the material balance equations for material j at time t. Then

$$RC_{jt}^{buy} = \Pi_{jt} - C_{jt}^{buy} \qquad (5.1)$$

$$RC_{jt}^{sell} = C_{jt}^{sell} - \Pi_{jt} \qquad (5.2)$$

$$RC_{jt}^{inv} = h_{jt} - \Pi_{jt} + \Pi_{j,t-1} \qquad (5.3)$$

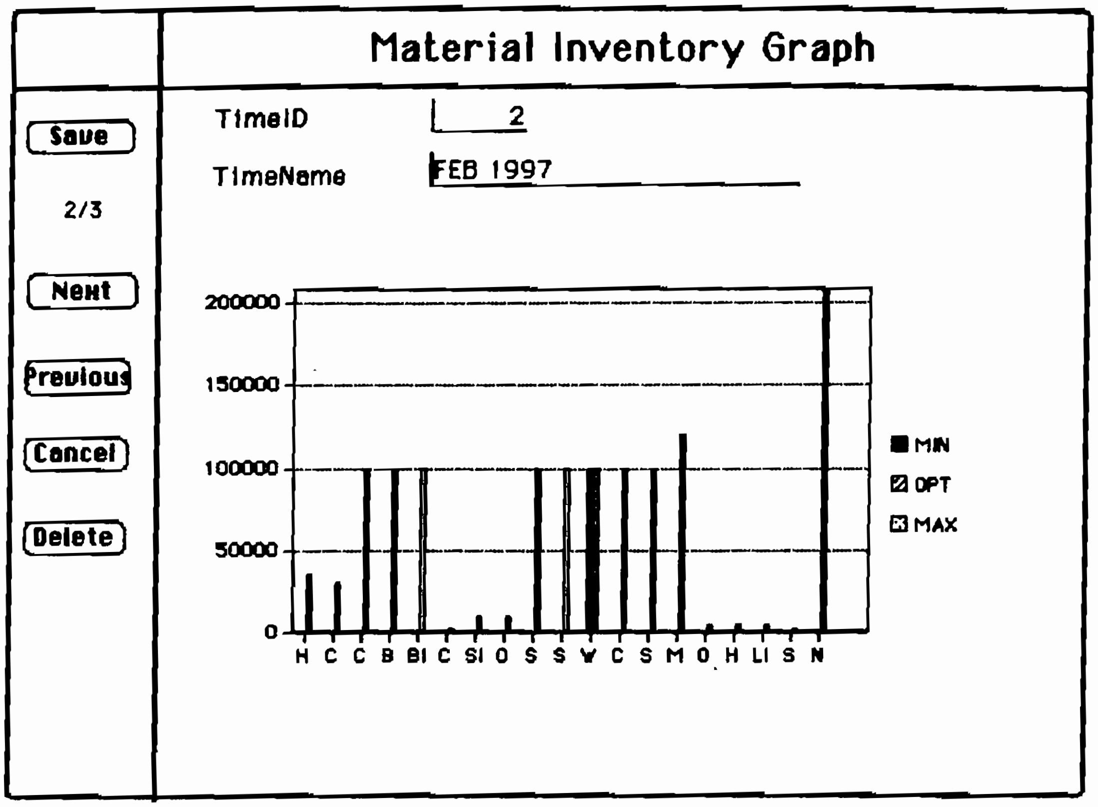


FIGURE 22 GRAPH OF MATERIAL INVENTORIED

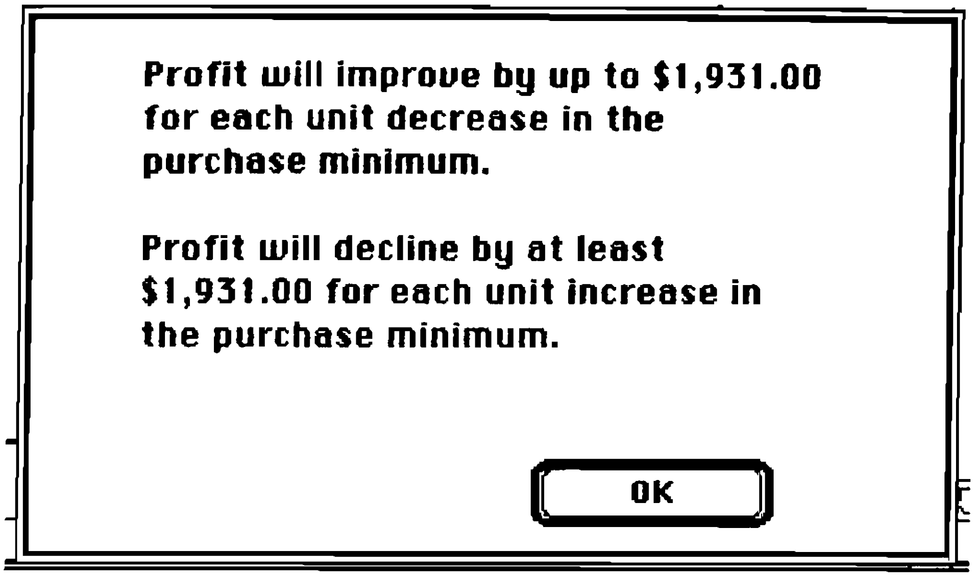


FIGURE 23 DISPLAY OF DUAL VARIABLES

Reduced costs are the dual values on the bounds. As the dual values on material balance constraints are available with the solution of the optimization problem, the reduced cost values of the variables (Material Bought, Material Sold and Material Inventoried) can be easily computed. The computation of reduced cost for material inventoried is slightly difficult as we need to store dual values for more than one time period, but we can use a global procedure and scripts to overcome this.

As we have explained earlier in sections 5.1 and 5.2, we can display the dual values (Figure 23) and the reduced costs in a layout for [Facilities] that contains a scrolling list of times or in a layout for [Times] in a scrolling list of facilities.

5.4 Optimal Summaries

In the case of a multi-period model, creation of summaries is a difficult and not straightforward like in single period models. In this section, we discuss two different ways the summaries can be displayed: summaries of each time period separately, and grand summaries for all time periods.

We repeat the equation of the objective function (equations of Appendix):

$$Z(t) = \Big(\sum_{J \in M} c^{sell}_{jt} x^{sell}_{jt} - \sum_{J \in M} c^{buy}_{jt} x^{buy}_{jt} - \sum_{(j,j')M^{conv}} c^{conv}_{jj't} x^{conv}_{jj't} -$$

$$\sum_{(i,k) \in F^{act}} c^{act}_{ikt} x^{act}_{ikt} - \sum_{j \in M} h_{jt} x^{inv}_{jt} - \sum_{i \in F} C^{vend}_{i,t} x^{vend}_{i,t} \Big) \qquad (5.5)$$

$$\mathbf{Z} = \boxed{\sum_{t \in T} Z(t)} \qquad (5.6)$$

We will now break it up into different parts. Typically a user would like to answer "What is the sum total of revenue obtained by selling all materials at one time (say Time = t)?" This figure can be obtained by searching for [Times]TimeID = t and summing over all the materials the quantity [Materials]MatTime'SellPrice multiplied by [Materials]MatTime'SellOPT. This will indicate the revenue obtained from the sale of all the materials at this time. Let us define it as it R(t), the revenue at time t :

$$R(t) = \sum_{J \in M} s^{sell}_{jt} x^{sell}_{jt} \qquad (5.7)$$

Similarly we can write the corresponding summation terms for the other terms. We define

$Cp(t)$ = Cost of purchase of all materials at time t

$Ca(t)$ = Cost of all activities at time t

$Ci(t)$ = Cost of carrying inventory at time t

$Cc(t)$ = Cost of conversions at time t

$Cv(t)$ = Cost of outsourcing at time t

Once we have calculated all the six quantities we can rewrite the net profit as the following:

$$Z(t) = R(t) - Cp(t) - Ca(t) - Ci(t) - Cc(t) - Cv(t) \qquad (5.8)$$

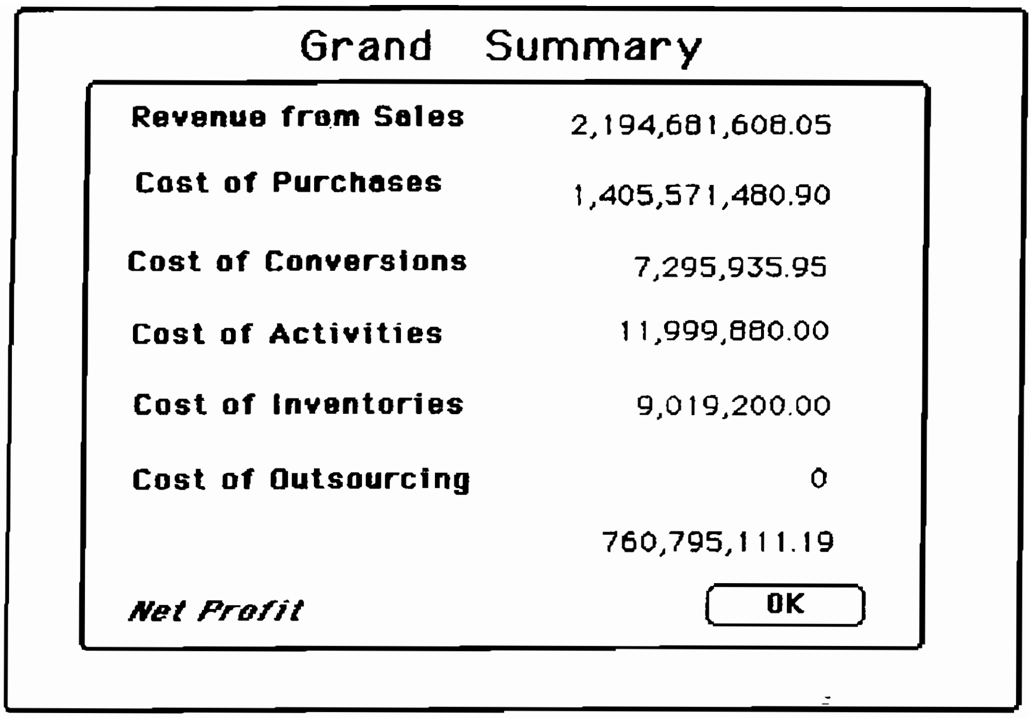


FIGURE 24 GRAND SUMMARY

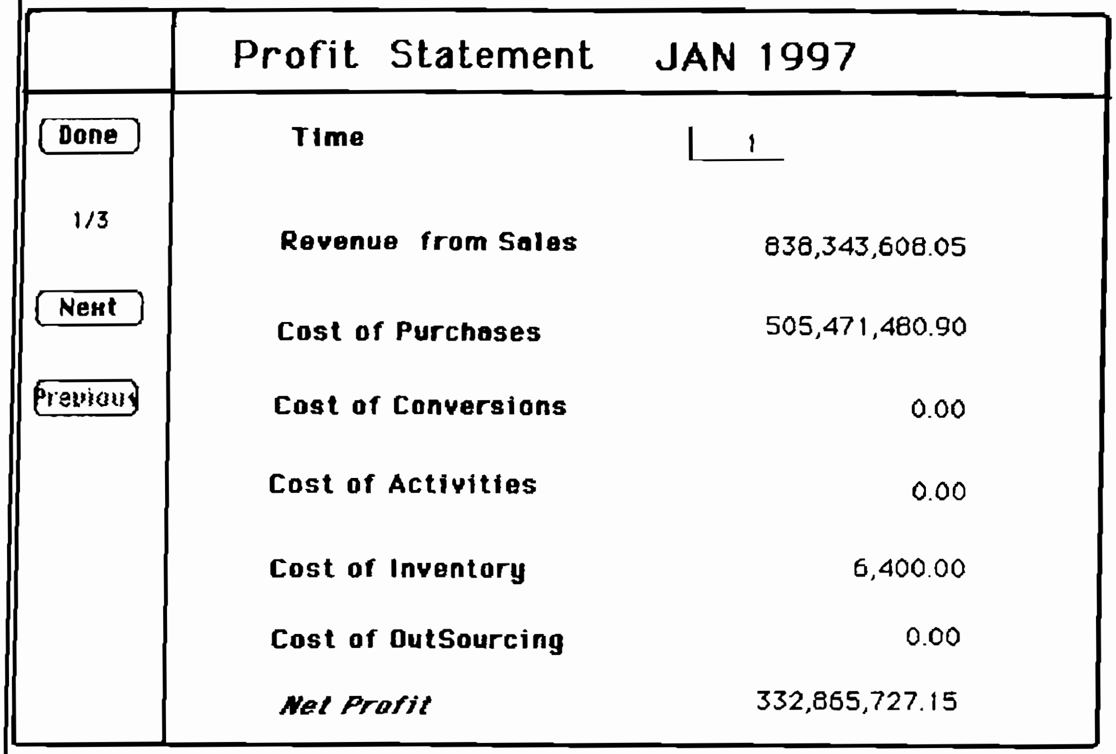


FIGURE 25 PROFIT STATEMENT OF ONE TIME PERIOD

The terms of equation of 5.5 can be displayed in a grand summary over all time periods (Figure 24). Based on the equation 5.8, we can also display the summary for each period (Figure 25).

5.5 Discounted Cash Flow and Capital Budgeting Issues

The advantage of the multi-period model is that we can incorporate the time value of money. In a financial analysis if there is no time value of money, we call the results a nominal cash flow. In a discounted cash flow, the user can choose the interest rate. The summary statement for each time and the grand summary statement can be converted to the discounted cash flows (Figure 26) and discounted summaries (Figure 27).

With the features of the 4th Dimension Package, we can have any one of three following:

1. Optimize with the nominal objective function and displaying the optimal result as a nominal cash flow having no consideration of discounting.

2. Optimize with the discounted objective function and display the optimal result in a nominal cash flow statement. In this case optimization is performed after discounting.

3. Optimize with the nominal objective and convert the optimal result to discounted cash flows and show the discounted cash flows. In this case the discounting is done after the optimization is performed.

Discounted Grand Summary

Revenue from Sales	2,084,107,383.89
Cost of Purchases	1,349,871,046.20
Cost of Conversions	0.00
Cost of Activities	11,421,658.53
Cost of Inventory	8,584,608.11
Cost of Outsourcing	0.00
Net Profit	714,230,071.03

OK

FIGURE 26 DISCOUNTED GRAND SUMMARY

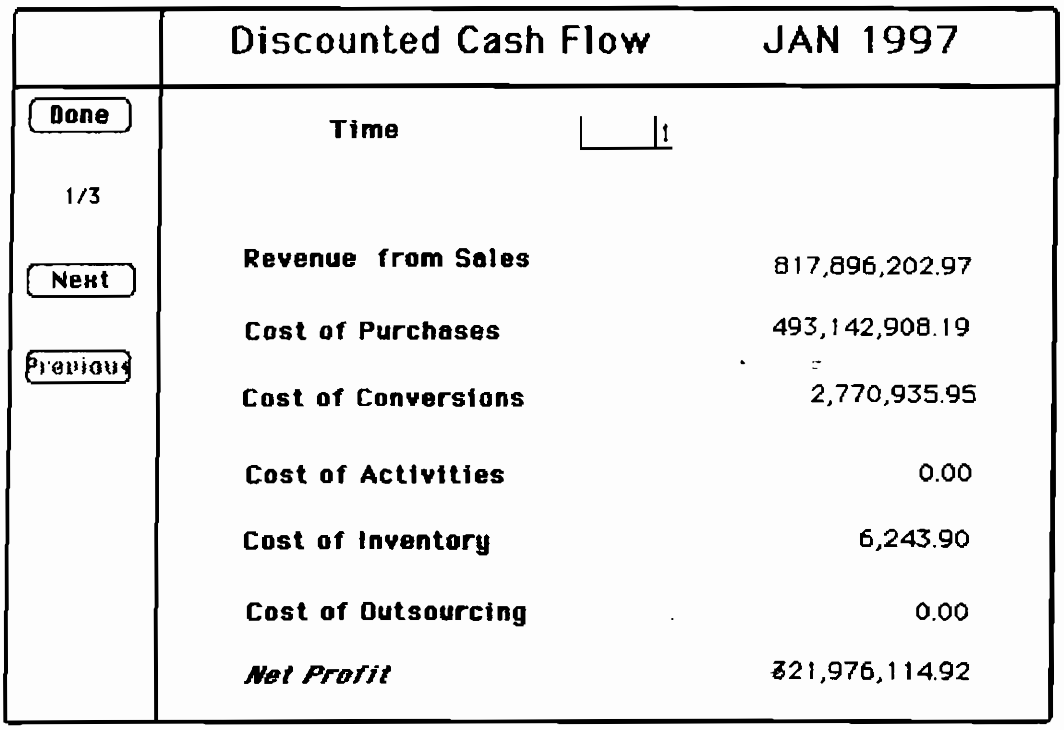


FIGURE 27 DISCOUNTED CASH FLOW

In 4th Dimension, we create a global procedure Discount. If the variable generation procedure discussed in 4.3.2, the objective function coefficients that represent nominal accounting figures (BuyPrice, Sell price, Cost of activity, Cost of material etc.) are changed to discounted objective function coefficients.

In the case of discounting, the unit of time is very important. If we are planning production for the short term, we need not worry too much about the interest rate as there will not be too much of a difference between of the optimal result of the discounted and nominal objective function. However, if we are using this DSS for long term strategic planning then discounting is very important.

We next discuss capital budgeting issues. In a previous study (Sihna and Dutta, 1985), we have shown how a system dynamic simulation model can be used for capital budgeting issues regarding the investment decisions of a steel plant (particularly of blast furnaces).

We will show how a multiple time linear programming based DSS can be used for capital budgeting issues in steel companies. As an example, we consider an investment proposal of X dollars in a steel plant with Blast Furnace, Continuous Caster, Hot Rolling Mill, Bar and Rod Mill and Sheet Mill. We assume that the Blast Furnace and Bar and Rod mill are the two bottlenecks in the system. In other words, when we run the optimization model, we find that "Facility Capacity" of the two facilities is at their upper limits. Therefore, the dual variables of the "Facility Capacity" constraint will indicate the amount by which the profit will increase if the capacity is increased by one unit (or one hour). In a blast furnace the production can be increased by increasing the available hours or by increasing the productivity. Now we assume that we are considering the introduction of a coal injection facility in the blast furnace, which would, increase the productivity of the blast furnace by 10 percent. After the coal injection is introduced, the yield and the productivity would increase (as the same hot metal is obtained from less iron-bearing material or sinter) we can change the parameters in the database accordingly.

So the model can be run with two different cases each with different variable cost or [Activity]ActCost and the grand summary (with discounted cash flow) can be examined. Based on this, we can determine whether the investment is justified. We must understand that in these cases, we are using the results from discounted cash flow analysis.

5.6 Update Issues

In the single period model, we have found that the time taken by the model to generate the constraints and variables is much longer than the time to solve the model. A study reported in Fourer (1997), shows that a problem with 857 constraints and 853 variables takes 12 minutes to write and generate constraints but 1.59 minutes to solve. In our DSS, using a 75 MHz machine, a problem (with 2838 variables and 1909 constraints), takes 98 minutes for generating and writing constraints and variables, but only 2.2 minutes to solve. Therefore, if the user wants to update one price for one period on one material, he or she should not go through the whole process of generating constraints, generating variables, writing constraints and writing variables.

Where there is no addition or deletion of records in the database, there are not many changes between scenarios. A substantial part of the user's time is spent in waiting for the generation stage to be completed. When two successive scenarios are similar, so are their corresponding linear programs. In most cases, changing one data value is the same as changing the lower bound, upper bound or coefficient in the linear program. The *Update* mode helps in this regard.

The following changes in the database may result in the changes in the linear programming model:

1. Addition of records in the Materials, Facilities, Activities, Times and Storage-Areas file.

2. Deletion of records from the above mentioned files.

3. Changing the names of the Materials, Facilities, Activities Storage-Areas or Times.

4. Changing the values of the numerical data in the fields and subfields of the database.

In case 1 and 2, when there are an additions and deletions of records in the five files, the numbers of variables and constraints in the database are going to change. In case 3, simply changing the name of any material, facility, activity or storage-area will not make the linear program different and there is no change in the optimal solution.

So we are left with Case 4. In this case only some numerical values in the database change, which requires corresponding changes in the relevant records and subrecords in the [Constraints] and [Variables] file. The Update mode helps in Case 4 only.

In a single period model, using an update mode allows the user to change the lower bound, upper bound, and the coefficients without creating/deleting any more records/subrecords. The 4th Dimension software allows us to make an update mode (Figure 28) that automatically makes the corresponding changes in the [Constraints] and [Variables] files. In the case of a multi-period model most of the data for the database are kept in the subfiles. Therefore, an update mode is harder to implement in the multi-period mode; our implementation has to extend the concept of the update mode to all files and subfiles of the database.

Here is how an update mode works. Say the Buy Price of the Material COIL is changed from $533 to $563 per ton at Time=2. The user goes to the *Update* option in the Menu and selects the [Materials] file and double clicks on the Material = COIL. He or she will get a layout on the screen called Materials Update. In this layout there is an included layout of Time which shows the values of some of the subfields of [Materials]MatTime. Then the user double clicks at the Time = 2 and gets screen layout which displays all the numeric value of all parameters of Material = COIL at this time. One of the parameters is the BuyPrice and the current value is $533. The user changes the value of BuyPrice from 533 to 563. As soon as the new value of BuyPrice is entered, a built-in program searches for [Variables]Type = Material Bought at [Variables]ID3= 2 (or Time=2) at [Variables]ID1=COIL. There should be one and only one value of this record in the [Variables] file. Then the [Variables]Objective value is changed from -533 to -563. Then the user clicks the three buttons to save the record in the [Materials] and [Variables] file and returns to the main menu. Now the user can write the linear program without re-generating the constraints and variables files.

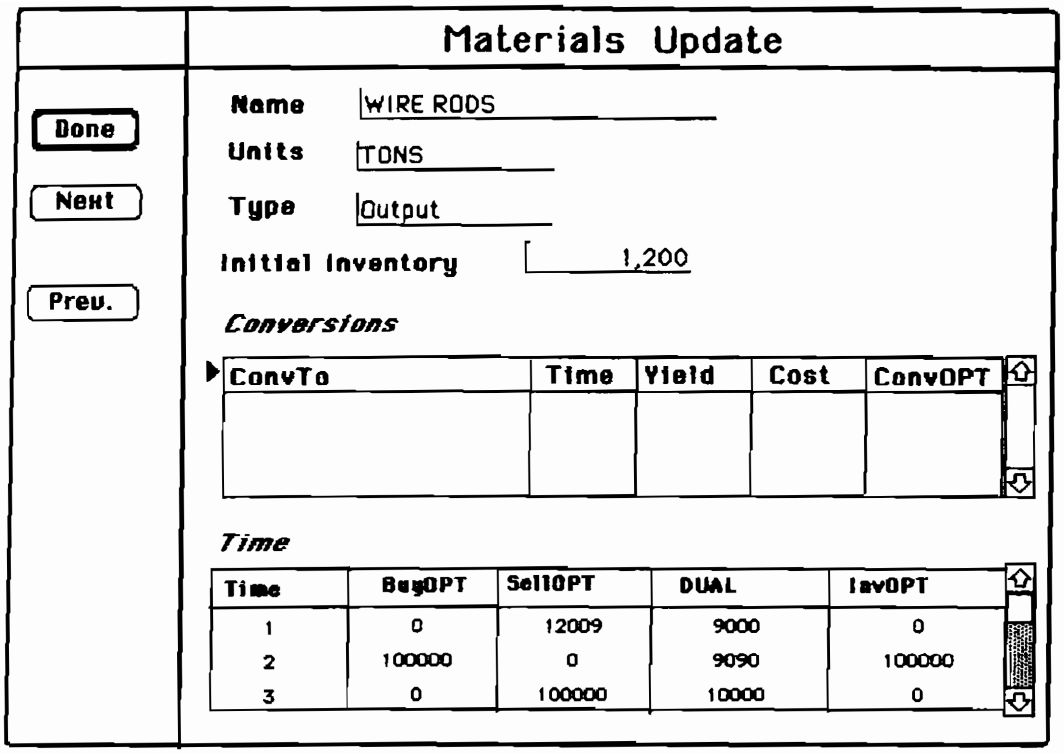


FIGURE 28 MATERIALS UPDATE

6. Comparison of Database Structures

In this section, we consider the different variations of the [Materials] and [Facilities] files. These files can be organized in several ways and we discuss how the computer times for variable and constraint generation vary with different variations of the relational and hierarchical databases. We show two different types of structures: STEEL-TIME1(Figure 29) and STEEL-TIME2 (Figure 30). The structure of STEEL-TIME1 is similar to STEEL-TIME (Figure 2) which we have discussed in Section 3. Fourer (1997) has studied two different variations of the [Constraints] and [Variables] files, one relational and one hierarchical. We extend his comparison to two different variations of the [Materials] and [Facilities] files. We compare the implementation of STEEL-TIME1 and STEEL-TIME2 according to four different criteria: ease of use, data storage and retrieval, ease of development and efficiency of optimization.

6.1 Implementation of STEEL-TIME1 vs. STEEL-TIME2

STEEL-TIME1 is a modified version of STEEL-TIME. STEEL-TIME1 and

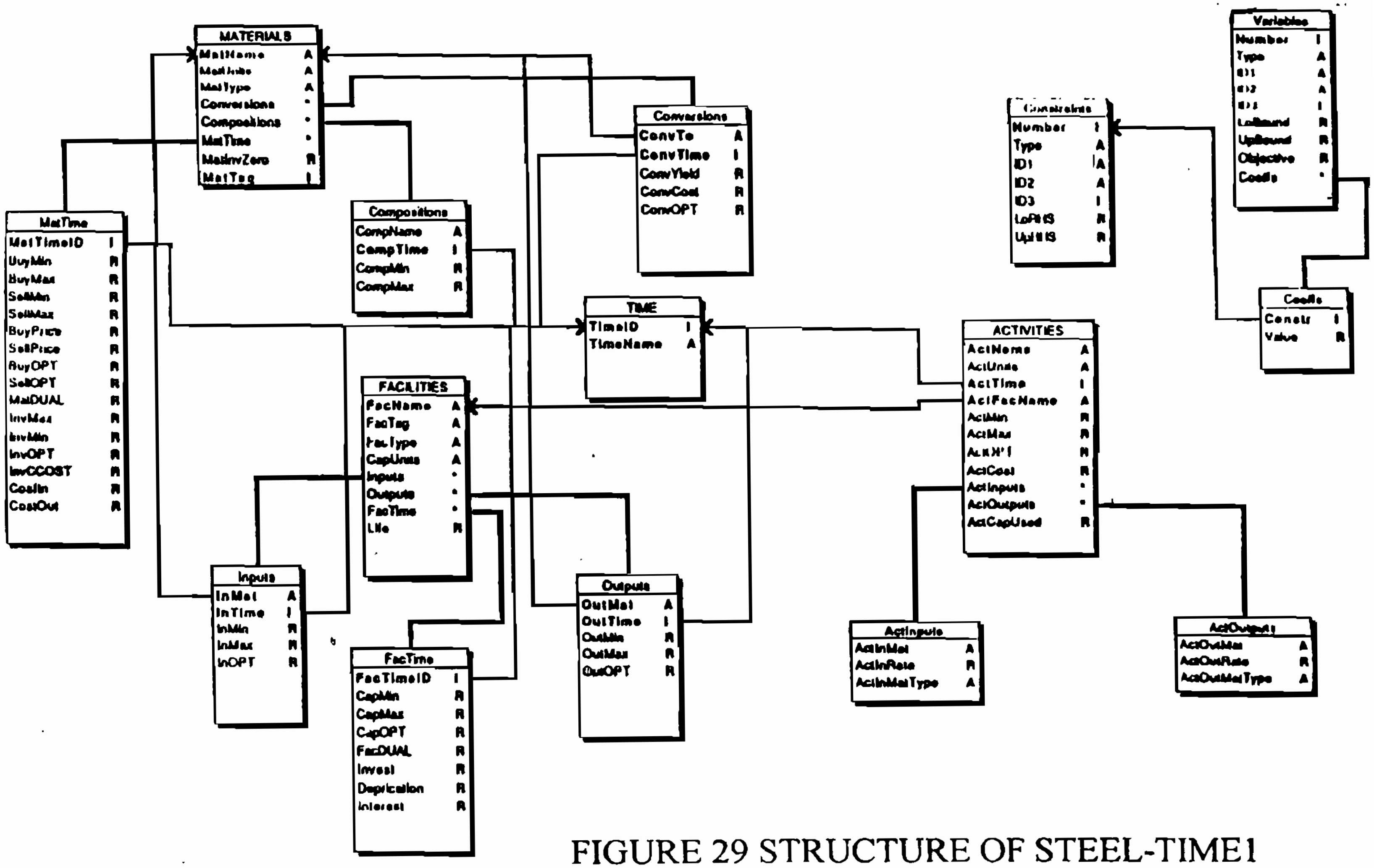


FIGURE 29 STRUCTURE OF STEEL-TIME1

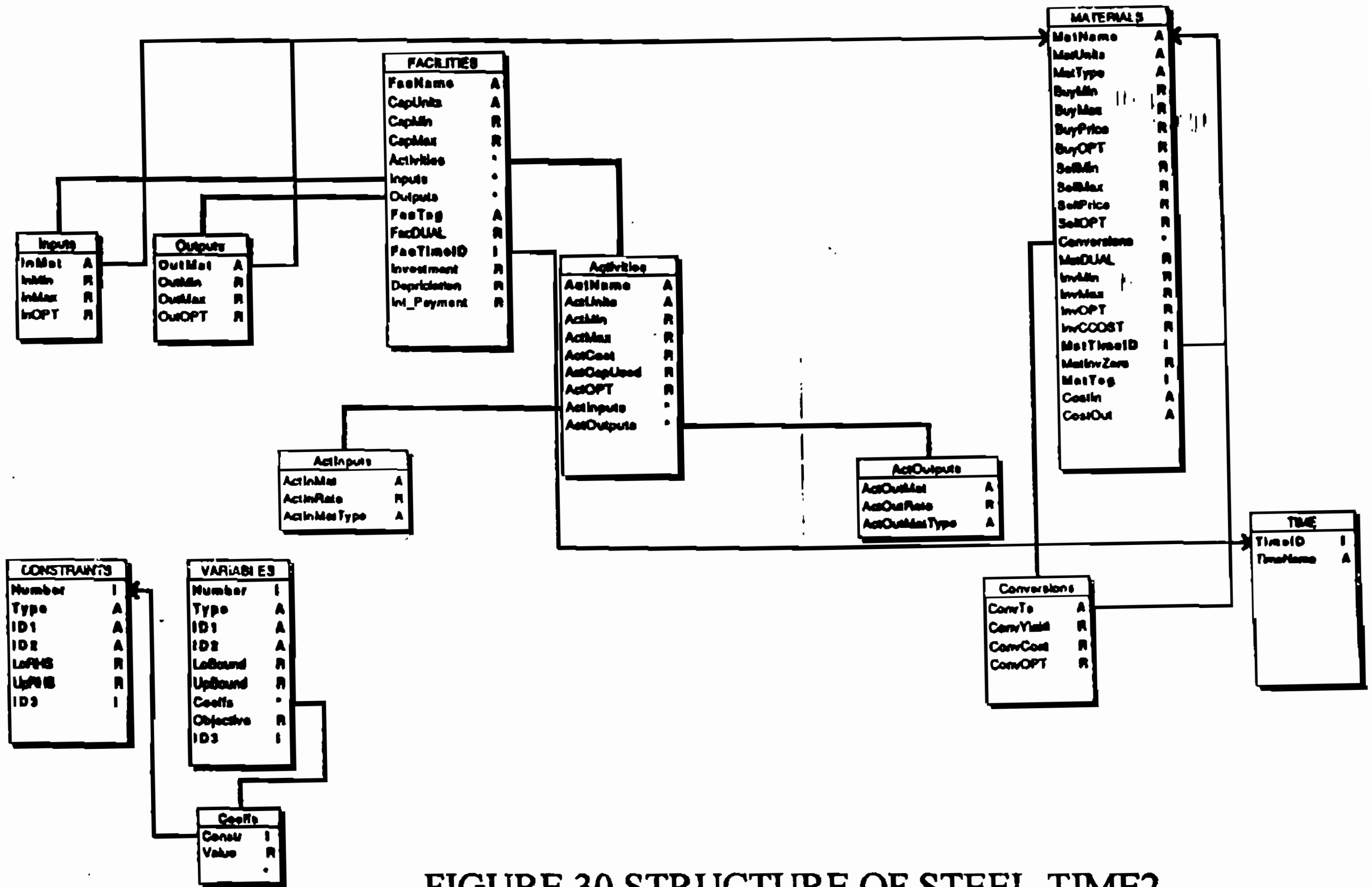


FIGURE 30 STRUCTURE OF STEEL-TIME2

STEEL-TIME are practically identical except that the file [Storage-Areas] is not present in STEEL-TIME1. The other difference is that the vendoring/outsourcing variable (or soft capacity variable) in the "Capacity Balance" constraint is not considered in STEEL-TIME1.

The numbers of constraints and variables in STEEL-TIME1 and STEEL-TIME2 are equal. The other similarities and differences of STEEL-TIME1 and STEEL-TIME2 are as follows:

1. In STEEL-TIME1, the time dependent parameters are in subfields of the [Materials] and [Facilities] files. In STEEL-TIME2 these are in the fields of the [Materials] and [Facilities] files.

2. The [Storage-Areas] file of STEEL-TIME is not considered in this comparison. In addition to that, vendoring or outsourcing is not considered an option. Even if the indexing in the formulation and the way of representing the mathematical model are different, we essentially solve the same optimization problem in STEEL-TIME1 and STEEL-TIME2.

3. STEEL-TIME1 or STEEL-TIME2 cannot be clearly classified as a purely relational or purely hierarchical database. Each has both relational and hierarchical aspects. STEEL-TIME1 is more relational and [Activities] is a separate file. STEEL-TIME2 is more hierarchical, and [Activities] is a subfile of the [Facilities] file.

6.2 Ease of Use

STEEL-TIME1 appears to be more complicated than STEEL-TIME2. Other than the [Times] file there are only two files in STEEL-TIME2, the [Materials] and the [Facilities] file. Therefore it is easier to use STEEL-TIME2 than STEEL-TIME1. In the [Materials] file, all the purchase, sales and inventories related data about the Materials are kept at the file level. When the materials are displayed on an output layout, in STEEL-TIME2, sorting is possible with respect to the [Materials]MatName as well as [Materials]MatTmeID. However in STEEL-TIME1, [Materials]MatName is at the file level and the [Materials]MatTime'MatTimeID is at the subfile level. So sorting is not possible at the same level in STEEL-TIME1.

In STEEL-TIME1, there are three files and [Activities] is a separate file related to the [Facilities] file. From a developer's point of view STEEL-TIME1 is more complicated than STEEL-TIME2. Moreover, most of the searches are performed at the sub-file level. For example, it is possible to list the dual prices and the reduced cost coefficients in the output layout at the file level in STEEL-TIME2, but similar lists are not possible in the STEEL-

TIME1. Such a display can be available in STEEL-TIME1 at the subrecord level only. On the other hand, STEEL-TIME1 has a greater flexibility for listing the activities, as [Activities] is a separate output file. Because of the inherent advantages of the relational file, the user will be able to update activities separately. Although we have not implemented this concept in STEEL-TIME1, such an implementation is possible. STEEL-TIME1 will also allow the user to compare two activities of two facilities by listing activities on the output file. So an activity PRODUCTION OF ES1 in three facilities M1, M2, M3 can be listed by performing a search with [Activities]ActName = "PRODUCTION OF ES1". Such searches are not possible with STEEL-TIME2.

6.3 Data Storage and Retrieval

STEEL-TIME1 satisfies the conditions of normalization that no piece of information be stored in more than one place. This condition is not satisfied in STEEL-TIME2. We also see that STEEL-TIME2 takes greater storage space than STEEL-TIME1.

In STEEL-TIME2 certain fields are repeated. [Materials]MatName, [Materials]MatType, [Materials]MatInvZero, [Materials]MatUnits, [Facilities]FacName and [Facilities]FacUnits are the fields that are repeated for every record of the [Time]TimeID file. This certainly requires more space for data storage, but does not pose a very serious problem with respect to ease of use. The 4th Dimension software allows a script to be written so that when the user enters the data for [Materials]MatName for one time period, the same [Materials]MatName is also available in other time periods. Therefore, as long as we are not changing [Materials]MatTime, we really do not need to enter the data for each time period.

6.4 Ease of Development

STEEL-TIME2 is easier to develop than STEEL-TIME1. However, we have decided to opt for STEEL-TIME1 as our main implementation, primarily because the latest version of the 4th Dimension software does not support more than one level of subfile. Because of the inherent advantage of relational databases, [Activities] was defined as a separate file in STEEL-TIME1, whereas it was a subfile in the [Facilities] file of STEEL-TIME2.

6.5 Efficiency

The times for constraint generation, variable generation and solution, and reading optimal values and the dual values are as shown in Table 5.

Table 5

Comparison of Steel-Time 1 and Steel-Time2

Computer ---------->		Macintosh II		Macintosh IIci		PowerMac 7200/75 MHz	
Data base---------->		STEEL-TIME1	STEEL-TIME2	STEEL-TIME1	STEEL-TIME2	STEEL-TIME1	STEEL-TIME2
Files	Units						
Materials	Records	19	57	19	57	19	57
Facilities	Records	7	21	21	21	21	21
Activities	Records	24	24 (Subfile)	24	24 (Subfile)	24	24 (Subfile)
Times	Records	3	3	3	3	3	3
Constraints	Records	141	141	141	141	141	141
Variables	Records	266	266	266	266	266	266
Disk Space (Model)	KB	688	336	688	336	688	336
Disk Space (Data)	KB	472	484	472	484	472	484
Cons. Generation Time	Seconds	16	16	42	39	12	12
Var.Generation Time	Seconds	169	88	328	172	109	45
Writing Constraint Time	Seconds	9	8	23	14	7	7
Writing Variable Time	Seconds	30	28	22	14	22	21
Solving	Seconds	8	8	8	8	8	8
Reading Optimal Value Time	Seconds	26	26	67	68	21	21
Reading Optimal Value Time	Seconds	12	12	47	28	8	8

We find that STEEL-TIME2 is faster in generating the variables and constraints than STEEL-TIME1.This is because in STEEL-TIME1, the data for time dependent parameters are stored in a subfile. So every time a record is written in the [Variables] file, first the record of the [Materials] is searched for, then the subrecord of the file is searched for, and then the record is written in the [Variables] file. However in STEEL-TIME2 fields like BuyMax, BuyMin are at the field level. Therefore to write a record in the[Variables] file, we only have to search the [Materials] and [Facilities] at the file level. Similarly, the disk-space for the data of STEEL-TIME2 is higher than that of STEEL-TIME1. This is because time-independent parameters like MatName, MatTag, MatInvZero, FacName, Factag, FacType etc. are duplicated in STEEL-TIME2.

After a careful comparison of these two variations, we find that STEEL-TIME2 is superior to STEEL-TIME1 on an overall basis. However we need to extend the present study so that STEEL-TIME2 is normalized. This can be done by replacing all the subfiles by files so that [Materials]MatTime and [Facilities]FacTime and other subfiles will be normalized with additional indices and key-fields. We will be in position to recommend STEEL-TIME2 only after that.

7. Extension and Conclusion

An extension of the DSS will be non-linearity of the model. Most of the industrial cost curves are non-linear or at best can be represented as having a piece-wise linear behavior. It will be interesting to study how to represent these non-linearities while retaining the model's user-friendliness.

A second extension of the model will be to have multiple objective linear programs and represent them in the database. This can be done by changing the model management system. For example, the current model can be changed to cost minimization, revenue maximization, maximization of marketable products (revenue or production), maximization of the utilization of the facilities etc. It is possible to have a menu driven program in this DSS which optimizes over different objectives.

A interesting extension will be to study the paradigm neutrality (Geoffrion, 1989) of this data structure for the multiple period model. Although the model is designed for the mathematical programming paradigm, we can extend it for inventory control and also for scheduling, vehicle routing and queuing applications. We have parameters for all materials at all times. We can determine the ordering and holding cost for all material and hence try to find optimal order quantities. However, the batch size will be decided by practical consideration like the heat size of the steel making shop, the capacity of the vehicle carrying the products and the capacity of the loading and unloading facility. Given that we have the batch size and lead-time of all materials produced the present model can be extended to a scheduling model of each product in each time.

Appendix-1

Optimization Model

As explained earlier, the model has the five fundamental elements: Materials, Facilities, Activities, Storage-Areas, and Times. The model is a generalized network flow model with the objective of maximizing the net profit the company. The user has a choice of changing the objective function from maximizing the net profit of the company to maximizing the net discounted profit of the company. The optimization is performed with the following constraints:

1. Material Balance
2. Facility Inputs
3. Facility Outputs
4. Facility Capacity
5. Storage Capacity
6. Storage Total

In addition to the above constraints, the each variable of the model (like the amount you can buy in particular period) is bounded by a upper bound and a lower bound.

A.1 Times Data

T is the set of times planning periods indexed by t

N= Planning Horizon

T = {1, 2, 3, …N}

int = The interest rate between two time periods.

A.2 Materials Data

M is the set of materials indexed by j

A is the set of chemical constituents (like C, Si, Fe, FeSi etc.) indexed by α

l_{jt}^{buy} = the lower limit of purchases of material j for each j $\in M$ and t $\in T$

u_{jt}^{buy} = the upper limit on purchases of material j for each j $\in M$ and t $\in T$

c_{jt}^{buy} = the cost per unit of material j purchased for each j $\in M$ and t $\in T$

l_{jt}^{sell} = the lower limit on sales of material j for each j $\in M$ and t $\in T$

u_{jt}^{sell} = the upper limit on sales of material j for each j $\in M$ and t $\in T$

c_{jt}^{sell} = the revenue per unit of material j for each j $\in M$ and t $\in T$

l_{jt}^{inv} = lower limit of inventory of material j for each j $\in M$ and t $\in T$

u_{jt}^{inv} = upper limit on inventory of material j for each j $\in M$ and t $\in T$

h_{jt} = holding cost of the material j at time t for each j $\in M$ and t $\in T$

I_{j0}^{inv} = initial inventory of the material j for each j $\in M$

$M^{conv} \subseteq M \times M$ is the set of conversions

(j,j') $\in M^{conv}$ means that material j can be converted to material j', and j # j'

$a_{jj't}^{conv}$ = number of units of material j' that result from converting one unit of material j for each (j,j') $\in M^{conv}$ and t $\in T$

$c_{jj't}^{conv}$ = cost per unit of material j of conversion from j to j' for each (j, j') $\in M^{conv}$ and t $\in T$

$M^{comp} \subseteq M \times A$ is the set of compositions

(j, α) $\in M \times A$ means Material j has constituent α for each j $\in M$ and $\alpha \in A$

$Comp_{\alpha jt}^{(\min)}$ = Minimum composition of the constituent α for each (j, α) $\in M^{comp}$ and t $\in T$

$Comp^{(\max)}_{\alpha jt}$ = Maximum composition of the constituent α for each $(j, \alpha) \in M^{comp}$ and $t \in T$

A.3 Facilities Data

F is the set of facilities indexed by i

l^{cap}_{it} = the minimum amount of the capacity of facility i that must be used for each $i \in F$ and $t \in T$

u^{cap}_{it} = the maximum amount of the capacity of facility i that must be used for each $i \in F$ and $t \in T$

$F^{in} \subseteq F \times M$ is the set of facility inputs:

$(i,j) \subseteq F^{in}$ means that material j is used as an input at facility i

l^{in}_{ijt} = the minimum amount of material j that must be used as input to facility i, for each $(i,j) \in F^{in}$ and $t \in T$

u^{in}_{ijt} = the maximum amount of material j that may be used as input to facility i, for each $(i,j) \in F^{in}$ and $t \in T$

$F^{out} \subseteq F \times M$ is the set of facility outputs:

$(i,j) \subseteq F^{out}$ means that material j is produced as an output at facility i

l^{out}_{ijt} = the minimum amount of material j that must be produced as output from facility i for each $(i,j) \in F^{out}$ and $t \in T$

u^{out}_{ijt} = the maximum amount of material j that may be produced as output from facility i for each $(i,j) \in F^{out}$ and $t \in T$

C^{vend}_{it} = the cost of vendoring (outsourcing) a unit of capacity of facility i at time t.

A.4 Activities Data

F^{act} is the set of activities indexed by k

$(i,k) \in F^{act}$ means that k is an activity available at facility i

l^{act}_{ikt} = the minimum number of units of activity k that must be run at facility i for each $(i,k) \in F^{act}$ and $t \in T$

u^{act}_{ikt} = the maximum number of units of activity k that may be run at facility i for each $(i,k) \in F^{act}$ and $t \in T$

c^{act}_{ikt} = the cost per unit of activity k at a facility i, for each $(i,k) \in F^{act}$ and $t \in T$

r^{act}_{ikt} = the number of units of activity k that can be accommodated by one unit of capacity at facility i for each $(i,k) \in F^{act}$ and $t \in T$

$A^{in} \subseteq \{(i,j,k,t): (i,j) \in F^{in}, (i,k) \in F^{act}$ and $t \in T\}$ is a set of activity inputs

$(i,j,k,t) \in A^{in}$ means that input material j is used by activity k at facility i at time t

a^{in}_{ijkt} = the number of units of input material j used by one unit activity k at facility i for each $(i,j,k,t) \in A^{in}$

$A^{out} \subseteq \{(i,j,k,t): (i,j) \in F^{out}, (i,k) \in F^{act}$ and $t \in T\}$ is a set of activity outputs

$(i,j,k,t) \in A^{out}$ means that output material j can be produced by activity k at facility i at time t

a^{out}_{ijkt} = the number of units of output material j that can be produced by one unit of activity i at time t for each $(i,j,k,t) \in A^{out}$.

A.5 Storage-Areas Data

S is the set of Storage-Areas indexed by s

l^{inv}_{st} = lower limit of the material stored in Storage-Area s at time t for each $s \in S$ and $t \in T$

u^{inv}_{st} = upper limit of the material stored in Storage-Area s at time t for each $s \in S$ and $t \in T$

A.6 Variables

x_{jt}^{buy} = units of material j bought at time t for each j $\in$ M and t $\in$ T

x_{jt}^{sell} = units of material j sold at time t for each j $\in$ M and t $\in$ T

x_{jt}^{inv} = units of material j inventoried at time t for each j $\in$ M and t $\in$ T

x_{j0}^{inv} = inventory of the material j at time 0 for each j $\in$ M

x_{jst}^{inv} = units of material j inventoried at time t in storage s for each j $\in$ M, t $\in$ T and s $\in$ S

$x_{jj't}^{conv}$ = units of material j converted to material j' for each (j,j') $\in$ M^{conv} and t $\in$ T

x_{ijt}^{in} = units of material j used as an input by facility i for each (i,j) $\in$ F^{in} and t $\in$ T

x_{ijt}^{out} = units of material j produced as an output by facility i for each (i,j) $\in$ F^{out} and t $\in$ T

x_{ikt}^{act} = units of activity k operated at facility i for each (i,k) $\in$ F^{act} and t $\in$ T

x_{it}^{vend} = units of capacity vendored / outsourced at facility i at time t for each i $\in$ F and t $\in$ T

A.7 Objective Function

The objective of this model is to maximize revenue from sales, less the cost of purchasing, converting, running activities, vendoring and holding inventories over all periods of time.

$$Z(t) = \sum_{j \in M} c_{jt}^{sell} x_{jt}^{sell} - \sum_{j \in M} c_{jt}^{buy} x_{jt}^{buy} - \sum_{(j,j') M^{conv}} c_{jj't}^{conv} x_{jj't}^{conv} - \sum_{(i,k) \in F^{act}} c_{ikt}^{act} x_{ikt}^{act} - \sum_{j \in M} h_{jt} x_{jt}^{inv} - \sum_{i \in F} c_{it}^{vend} x_{it}^{vend} \tag{1}$$

$$Z = \sum_{t \in T} Z(t) \tag{2}$$

The first term in equation (1) is the revenue from sales. The second, third and fourth terms are the cost of purchasing, conversion and activities respectively. The fifth term is the inventory holding cost. The last term is the outsourcing cost. Equation (2) is the sum of Equation (1) over all periods of Time. Constraints The various constraints for this model are described next.

A.8 Constraints:

We now describe the various constraints

Material Balance

For all j $\in M$ and t $\in T$ the amount of material j made available by purchases, production and conversions and inventory must equal the amount used for sales , production, conversions and inventory:

$$x_{jt}^{buy} + \sum_{(i,j) \in F^{out}} x_{ijt}^{out} + \sum_{(j',j) \in M^{conv}} a_{j'jt}^{conv} x_{j'jt}^{conv} + x_{j,t-1}^{inv} =$$

$$\sum_{(i,j) \in F^{in}} x_{ijt}^{in} + x_{jt}^{sell} + \sum_{(j,j') \in M^{conv}} x_{jj't}^{conv} + x_{jt}^{inv} \qquad (3)$$

Facility Inputs

For each (i,j) $\in F^{in}$ and t $\in T$, is the amount of input j used at facility i must equal the total consumption of all activities at facility i:

$$x_{ijt}^{in} = \sum_{(i,j,k,t) \in A^{in}} a_{ijkt}^{in} x_{ikt}^{act} \qquad (4)$$

Facility Outputs

For each (i,j) $\in F^{out}$ and t $\in T$, the amount of output j produced at facility i must equal the total production of all activities at facility i:

$$x_{ijt}^{out} = \sum_{(i,j,k,t) \in A^{out}} a_{ijkt}^{out} x_{ikt}^{act} \tag{5}$$

Facility Capacity

For each i $\in F$ and time t $\in T$, the capacity used by all activities at facility i must be within the specified limits:

$$l_{it}^{cap} \leq \sum_{(i,k) \in F^{act}} x_{ikt}^{act} / r_{ikt}^{act} \leq u_{it}^{cap} + x_{it}^{vend} \tag{6}$$

Storage Capacity

For each s $\in$ S and time t $\in T$, the sum of all materials stored in storage-areas must be within the specified limits.

$$l_{st}^{inv} \leq \sum_{j \in M} x_{jst}^{inv} \leq u_{st}^{inv} \tag{7}$$

Storage Total

For each j $\in$ M and time t $\in T$, the sum of material j inventoried in all Storage-Areas must be equal to the total amount of that material inventoried.

$$\sum_{s \in S} x_{jst}^{inv} = x_{jt}^{inv} \tag{8}$$

Bounds

All variables must lie within the relevant limits defined by the data:

$$l_{jt}^{buy} \leq x_{jt}^{buy} \leq u_{jt}^{buy} \quad \text{for each } j \in M \text{ and } t \in T \tag{9}$$

$$l_{jt}^{sell} \leq x_{jt}^{sell} \leq u_{jt}^{sell} \quad \text{for each } j \in M \text{ and } t \in T \tag{10}$$

$$l_{jt}^{inv} \leq x_{jt}^{inv} \leq u_{jt}^{inv} \quad \text{for each } j \in M \text{ and } t \in T \tag{11}$$

$$0 \leq x_{jj't}^{conv} \text{ for each } (j,j') \in M^{conv} \text{ and } t \in T \tag{12}$$

$$0 \leq x_{it}^{vend} \text{ for each } i \in F \text{ and time } t \in T \tag{13}$$

$$0 \leq x_{jst}^{inv} \text{ each } j \in M,\ s \in S \text{ and time } t \in T \tag{14}$$

$$l_{ijt}^{in} \leq x_{ijt}^{in} \leq u_{ijt}^{in} \text{ for each } (i,j) \in F^{in} \text{ and } t \in T \tag{15}$$

$$l_{ijt}^{out} \leq x_{ijt}^{out} \leq u_{ijt}^{out} \text{ for each } (i,j) \in F^{out} \text{ and } t \in T \tag{16}$$

$$l_{ikt}^{act} \leq x_{ikt}^{act} \leq u_{ikt}^{act} \text{ for each } (i,k) \in F^{act} \text{ and } t \in T \tag{17}$$

Initial Conditions

$$x_{j0}^{inv} = I_{j0} \text{ for each } j \in M \tag{18}$$

A.9 Discounted Objective Function

The objective function can be changed to a discounted net profit maximization, by changing all cost profit parameters to discounted cost. If Zd is the discounted objective of nominal objective function Z (defined in 4.2), then the nominal and the discounted objective functions are related as follows:

$$Zd = \sum_{t \in T} Z(t)(1 + \text{int})^{-t} \tag{19}$$

References

Dutta G.;and Fourer R., 1996, "*A Survey of Mathematical Programming Applications In an Integrated Steel Plant,*" Abstract Booklet, IFORS-96, pp. 84.

Dutta G.; Sinha G. P; Roy P.N; and Mitter N., 1994, *"A Linear Programming Model for Distribution of Electrical Energy in a Steel Plant,"* International Transactions in Operational Research, Vol. 1, No 1. pp.17-29.

Fourer R., 1997, *"Database Structure for Mathematical Programming Models,"* Decision Support Systems, Vol. 20, pp. 317-344.

Geoffrion A. M., 1989, *"Computer-based Modeling Environments,"* European Journal in Operations Research, Vol. 41, pp. 33-43.

Marsten R.E., 1981, *"The Design Of XMP Linear Programming Library,"* ACM Transactions on Mathematical Software, Vol.7, pp. 481- 497.

Sinha G. P.; Chandrashekaran B.S.; Mitter N.; Dutta G.; Singh S.B.; Roy P.N.; and Roy Choudhary A., 1995, *"Strategic and Operations Management with Optimization at Tata Steel,"* Interfaces, Vol. 25, No. 1, pp. 6-19.