Virtual Data Warehouse Modeling Using Petri Nets for Distributed Decision Making

[1] Nabendu Chaki, [2] Bidyut Biman Sarkar
[1,] *University of Calcutta, Kolkata, India, nabendu@ieee.org*
[*2,] *Techno India, Salt lake, Kolkata, Inda,bidyutbiman@gmail.com*
doi: 10.4156/jcit.vol5.issue5.1

Abstract

A decision maker wants the pool of data at the finger tip while making the decision. In state of the art applications, decision making is no more a centralized process. Distribution of resources is a challenge before the system designers. Besides, for timely analyzing the distributed data, a robust query processing system along with the physical storage with schema definitions are also necessary. In the present state of business process skills, technologies, processes, applications and practices all are falling under the categories of competitive intelligence. In short BI aims to support better business decision-making. In this paper we propose an analytical model using Petri Net for distributed data management in a data warehouse to ease the OLAP (Online Analytical Processing) operations. Some of the properties of the model like safeness, boundedness, liveness and conservativeness are also verified..

Keywords: *Virtual Data Warehouse, Distributed Systems, Analytic Processing, Petri Net, Star Schema, Reachability*

1. Introduction

Business intelligence is used for fact based decision making. It is a data oriented technology and applied to carryout sustainable competitive advantage and core competence over the other competitors. BI applications require Data Warehouse as storage for historical data. It is used for decision making like, who could be the best supplier among the available vendors or analysis of the past performance before launching a product at a particular place or used for prediction of population growth of a country over a specific period of time and its proportional requirement of food, land, education, job etc. In the past some of the requirements were catered through tedious statistical analysis. A suitable model is necessary for computation of large size repositories.

Summarized data from functional data bases are the basic sources of data for Data Warehouse. Transaction processing is difficult when the repositories are distributed. Data in a Warehouse is non volatile in nature but it does not require data updating, transaction processing, recovery management and concurrency control. Data Warehouse is a multi dimensional data model viewed in the form of a data cube [1]. There are three different types of data warehouse models namely Enterprise Warehouse, Data Mart and a Virtual Warehouse. Enterprise Warehouse deals with all functional activities of the enterprise. Data Mart deals with some specific functional activities of the enterprise. Virtual Warehouse deals with operational data sources with limited transaction processing and data integration functions and also some summary views of the Data Warehouse According to the complexity of the problem the measuring dimensions of the model will also vary and so as the quires.

Due to intrinsic nature of the decision making processes, every instance of the transaction is of importance to the business house. Building a Data Warehouse model for historical repository is not the solution rather a generic framework for handling distributed repository along with transaction management constraints are necessary.

In this paper an attempt is made to design a generic scalable model of a Virtual Data Warehouse. Petri Net is an analytical tool for modeling concurrent, distributed, asynchronous, parallel, deterministic and stochastic system. We revisit the virtual Data Warehouse model with help of a Petri Net by constructing a Reachability tree to analyze safe, bounded, live and conservative properties of

the model and then use the technique of slicing the multidimensional Virtual Data Warehouse model to reduce the dimensional and time complexity of query computation in a non loss manner.

We deal with the design of a generic model of Virtual Data Warehouse in section 2. In section 3 a case study on inventory holding of an enterprise is presented to explain the functionalities of the model along with some critical queries. In section 4, we revisited the Virtual Data Warehouse model through Petri Net for Reachability analysis. In section 5 the concept of slicing-based approach to reduce the operational complexities of the VDW model along with few queries are described. Section 6 presents the concluding remark and plan for future work. List of references is presented in section 7.

2. Virtual Data Warehouse Design

Y. Zhao and his colleagues [12] describe a middleware service suite "GriPhyN" for virtual data analysis from a common multidimensional data model. The definition and query expressions are expressed in a virtual data language called VDL. Information about data and computational procedures are stored in a virtual data catalog called VDC. For various data intensive applications VDS is used. The virtualization is attempted with respect to location, representation, and materialization.

Data warehouse is a subject oriented, integrated, time-variant, non-volatile collection of data for corporate decision making [2]. Subject oriented indicates discussion about some subject chain like procurement, production and sales of a product of an enterprise. Integrated indicates data of a particular subject from all possible sources, irrespective of its homogeneity or heterogeneity in nature like package for heart operation includes: Operation Theatre charge, cost of Surgeon, Medicine, Pathological charges etc. Time variant means data about past events which provides information for the present instance. Nonvolatile indicates no transaction processing, no updating only data loading and accessing.

Data is stored in a relational database management system (RDBMS) or in some other form of data storage. Informational data a kind of history generated out of operational data is stored in data warehouse (DW). When a Data Warehouse is connected with operational data base through the use of middleware (soft wares acting as a interface between front end tools and backend servers) is called as virtual Data Warehouse [3]. Yuan Ji in 2001, in his report describes a framework for virtual data warehouse implementation for automatic translation of the data model into XML format. A Multidimensional Data Query language (XMDQL) along with different application program interfaces (API) is described. To categories the three different kinds of query on the warehouse like warehouse query, Cube Schema Query and cube data Query is described. However, Inmon criticized the significant disadvantages of virtual data warehouse approach [2] from the view point of performance level reduction, high volume of historical data source, unmatched data formats between OLAP and operational data, Non availability of automatic integration of legacy system and aggregation of summarized data of warehouse with detail data of operational database [4]. OLAP (Online Analytical Process) is a set of measures used to analyze the data of a Data Warehouse or a Data Mart. Contents can be viewed with the measures like; Roll-up, Drill-down, Slice, Dice and Pivot. Functional data is modeled and stored separately in a Data Warehouse. Data from a particular functional activity is called Data mart. 3-Tier architecture is practiced to enable step-wise refinements and the rules are designed as guidelines. OLAP is a set of measures used to analyze the data of a Data Warehouse or a Data Mart. The measures to view and study the systems dynamics Roll-up, Drill-down, Slice, Dice and Pivot are used.

2.1. Distributed DW Architecture

While designing the data warehouse model for distributed decision making the quality of the multidimensional association rule for meeting the optimum user requirement is considered to be one of the most important characteristics. The complexity of the association between the attributes depends on the attribute granules and their corresponding mappings at multi-tiers of the system [8].

Let us consider the hierarchy shown in figure 1 and combine the distributed data marts. We start at the apex level or the country level (a). Build a data mart for each county (b_1, b_2,….b_n) and subsequently

build the data marts for each province ($c_{11}…c_{13}$, $c_{21}…c_{23}$,….c_{n1}….c_{n3}), which may contain data from any application area starting from purchasing to sales or data related to finance of an enterprise. On the basis of county-level data marts, the provincial level data marts are built so that each provincial data mart holds data generalized from the county-level data marts located in the corresponding province. Finally, a nation-wide data warehouse is created over the provincial level data marts, and there is only one DW at the national level.

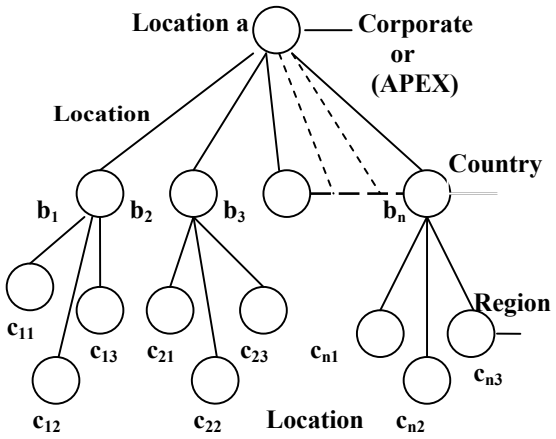

Figure 1. Concept Hierarchy in Operational Units

Therefore, a three-tier distributed DW architecture is created. The three tiers of the data warehouse are as follows:

Tier 1: Is the lowest layer where inputs are separated from operational databases and output to data marts. In particular, these refresh-operations are formulated by queries on the operational databases. The refresh operations can be expressed with the help of relational algebra or SQL.

Tire 2: Data warehouse itself with predefined schemas like Star or snowflake represents the tire 2. No complex transactions are allowed. Only write-operations are the refresh operations connect the warehouse to the operational databases. All other operations only read data from the data warehouse. It just builds views for the "data marts" or dialogue objects that are used as the OLAP interface. Thus the view construction operations are linked with the middle tier to the top tier, which deals with OLAP.

Tier 3: The top tier is constructed from the OLAP operations. Each user has a collection of data marts. At any time new user may enter into the system and create a new dialogue objects without closing the existing ones. Part of the functionality of the OLAP top tier deals with adding and removing users and data marts. In particular, if a user leaves the system, all data marts owned by the user are removed.

2.2. Conceptual Model of Data Ware House

The dimensions of the model are measured from the concept hierarchies. The derivable measures are discovered from the problem definitions and with the help of a star schema or a snow-flake schema the conceptual model is build. A schema is a collection of database objects, including tables, views, indexes, and synonyms. In star-schema a detail fact table is linked to dimension tables.

The Star Schema diagram graphically models the end-user's view. The diagram has three main components:

- Fact Table and its contents: metric attributes and the foreign keys necessary to join to the dimension tables.
- Dimension Tables and their contents: reference attributes, hierarchical attributes, and metric attributes.
- Dimension Tables are linked to the Fact Tables.

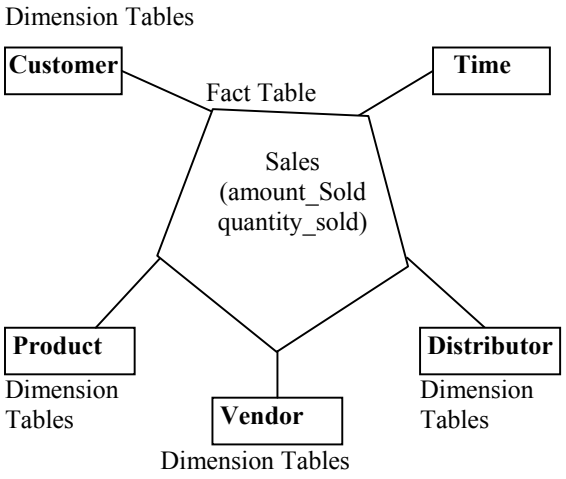

Figure 2. Star Schema

The snowflake schema is more complex than a star schema. Snowflake schemas attempts to normalize dimensions to eliminate redundancy. Let us now define the sales data cube and the dimensions of the sales cube customer, product, time, vendor and distributor in star schema using DMQL (Data Mining Query Language) of the above example and write some typical quires to extract data.

2.2.1. Cube operation

Define cube
Sales [product, customer, vendor, date, distributor]:
Total sold= sum (amount sold);
Qty_Sold = count (*) (1)

Define dimension customer as
(customer key, customer name, bank code) (2)
Define dimension product as
(product_key, product_Name, supplier_ID) (3)
Define dimension Time as
(Time key, date, month, Quarter, year) (4)
Define dimension Vendor as
(vendor_Key, vendor_ name, Loc, product) (5)
Define dimension distributor as
(distributer_ID, channel name, channel_charge) (6)

(1) defines the data-cube and (2) to (6) defines the fact dimensions.

As for example; let us assume a query on item sold by quantity and value at different dates by product and customer. With a simple SQL Query the following group-by computations needs to be performed:

(date, product, customer);
(date, product), (date, customer), (product, customer);
(date), (product),(customer);
( );

It will be highly inefficient. However, a simple DMQL query can be framed for efficient OLAP operations from the data cube is as follows:

Select product, time, customer, Total_Sold, Qty_Sold
From Sales

Cube By product, Time, customer;

3. Case Study

An export oriented lace manufacturing SME having turn over around 600 thousand million Bhat having a huge inventory holding of yarn as raw material and lace as finished product over four geographically dispersed locations in Thailand with its corporate office and marketing office at Bangkok The primary need is quantitative assessment of the item wise stock status at the work in progress (WIP) like stock of raw material, stock at quality control, stock at dyeing unit, stock at production and finished stock at the warehouse for improvement of operating efficiency. Due to non availability of timely assessment of stock status the marketing team at the customer point indicates a pessimistic date of delivery which affects the revenue of the company and sometimes order slips away. A huge stock clearance at the year end at a reduced cost and increased marketing expenses becomes the annoying factor to the board of the company. For strategic level decision making the most wanted information's are like:

1. Who are the most and least poised customers in last five years and what products are they buying?
2. Which product is a highest revenue generator and its stock status?
3. How revenue will be affected for setting up a new product or a new unit is absent from the system?
4. What could be the deciding factors for export promotions of products at different regions to enlarge the company turnover?

A Virtual Data Warehouse facilitates a single, integrated store of historical as well as operational data from all the distributed sources and locations compiled and made available to the corporate for the use of strategic decision making.

3.1. Data Model of the VDW

Due to rise in data volumes and operational complexities distributed computing is proposed for efficient distribution of the data at nodal points. We first require to plan and identify the places of nodes and chunks of data. The fragments stored on each node must be indexed in a uniform way in the lattice structure of the warehouse for effective interaction and optimized query evaluation plans [11]. The suggested abstracted model of virtual data warehouse VDW is a data cube presented in figure 3.

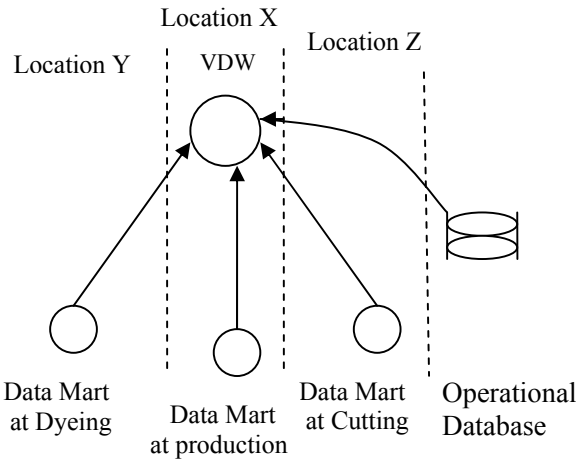

Figure 3. Distributed virtual data warehouse model

It is a virtual representation of the functionally distributed data marts available at the corporate and marketing unit at location X. The other functional data marts are available at dyeing unit at location Y and cutting unit at location Z.

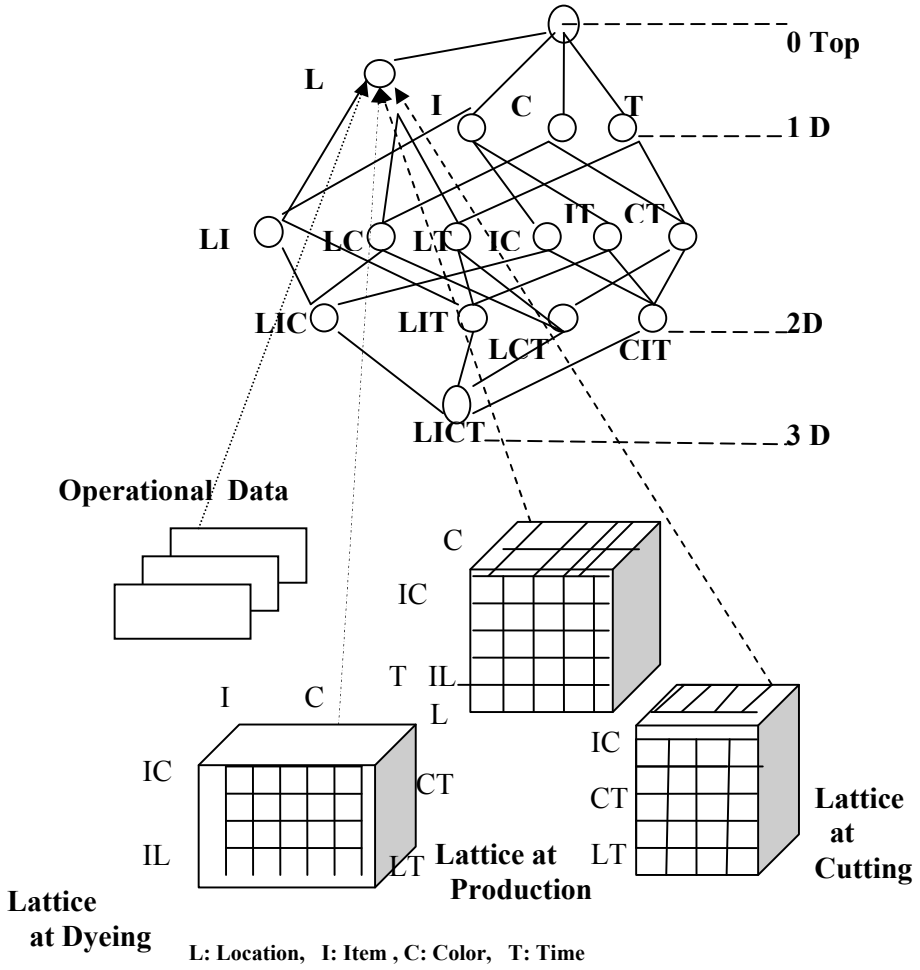

Figure 4. Lattice of cuboids of VDW

Operational data base is available at any location or in all locations but the diagram indicates only one operational data source at some location. In figure 4, the VDW is presented with its exact dimensions along with their distributed lattices at the respective locations. The objective of this diagrammatic presentation is to indicate that all OLAP operations at the marketing or at the corporate will be performed from the VDW and the local OLAP operations are performed from the respective data marts. Mapping between VDW, Data Mart and the operational data is through the location dimension. Data loading to the VDW as and when required is executed through the location dimension of the VDW. The tools for loading the data are DMQL. There can be three types of query warehouse query, cube schema query and cube data query [3]. Warehouse query is used by the user for the first time to get the information like warehouse name, warehouse description, and number of data cubes and the size of each cube. Cube schema query is used by the user to invoke the data dictionary of a particular cube to know the cube description, dimension and measurement. Cube data query is used to obtain a multi dimensional cube or any subset of it. This is particularly useful in applications for distributed decision making. Complexity of the model will increase if the operational locations are increased like quality control department start functioning as a separate cost centre or export

processing needs to setup at a new location. In such a situation the number of data marts and operational data sources will increase and so as the data loading and distributed query processing time complexity will also increase [5].

Some of the other critical issues of decision making on a distributed environment are unique legacy system with the data of the respective location and its model. However, the decision making is not an exclusive event. At the time of decision making the models of all the locations should be executed in sequence and composed off to take the decision.

4. Petri Net Model

Petri Net [6] is one of the most widely used modeling and analysis tool. The classical Petri net is a directed bipartite graph .The model describes the states, events, conditions, choice, iterations and parallelism. The two types of nodes are called places and transitions. Places and transitions are connected via arcs. Places are graphically represented by circles, transitions by bars. Places can store tokens, represented by black dots. A distribution of tokens on the places of a net is called a marking, and corresponds to the ``state'' of the Petri Net. A transition of a net is enabled at a marking if all its input places contain at least one token. An enabled transition removes one token from each of the input places, and adds one token to each of its output places. This is called the firing rule. Formally, a Petri Net is described by the four-tuples.

$PN= (P, T, D-, D+)$ (1)

where: P is the set of places and $|P|=m$.

T is the set of transitions and $|T|=n$

D^- :$P\times T\rightarrow N$ is the pre incidence matrix that specifies the arcs directed from places to transitions.

D^+:$T\times P \rightarrow N$ is the post incidence matrix that specifies the arcs directed from transitions to places.

As for example, let us try to model the question answer session of a visual reality show, where a single question ‘A’ is asked to two participants ‘B’ and ‘C’. The first answered question will move to ‘D’. A token at ‘E’ indicates a correct answer of the question asked otherwise no token.

The four transitions t_1, t_2, t_3 and t_4 describes as t_1 question fires from A and the same copy move to B and C in parallel as a synchronous event. Firing from t_2 indicates answer from B, t_3 answer from C. At D only one answer will move which ever answers first i.e. $\{ti>tj, i\neq j\}$, where i, j are two different events. At t_4 answer from ‘D’ (either of B and C) and the correct answer option from ‘A’, will appear as an input and a token or no token at E indicates correct answer or wrong answer from the participants.

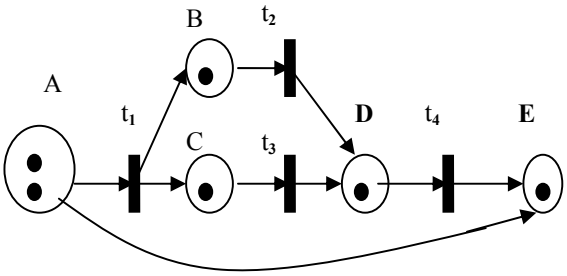

Figure 5. Reality show in operation

The markings at the respective locations are:

	A	B	C	D	E
Initial (Idle)	[2	0	0	0	0]^T
Current (working)	[1	1	1	1	0]^T
Final(on correct answer)	[0	0	0	0	1]^T

4.1. Petri Net Model of the VDW

In figure 6, a Petri Net model has been shown for a virtual data warehouse. Place P_8 has been used to model receipt of data from the production location modeled by place P_2, the dyeing location modeled by place P_4, and the cutting location in place P_6. The three transitions t_2, t_5 and t_8 respectively connects places P_2, P_4 and P_6 to P_8. The finished stock from P_7, as well as the order data and the sales data arrives from the places P_9, and P_{10} respectively. Availability of all these resources triggers the arrival of stock in the virtual data warehouse. This has been modeled using the transition t_9 and the VDW is organized. Queries related to Sales, Order and Stock are processed using the places P_{11}, P_{12} and P_{13} respectively. The Petri Net model is with 13 places (P_1 to P_{13}) and 10 transitions. Production process initiates at (P_1 --- P_2), (P_3 --- P_4) is the dyeing process, (P_5 --- P_8) cutting process are parallel processes and distributed in nature. The place P_8 in the model represents the Virtual data warehouse, while places P_{11} through P_{13} represent the slice based queries from the VDW. The place and transition functions at the respective locations have been described in Table 1.

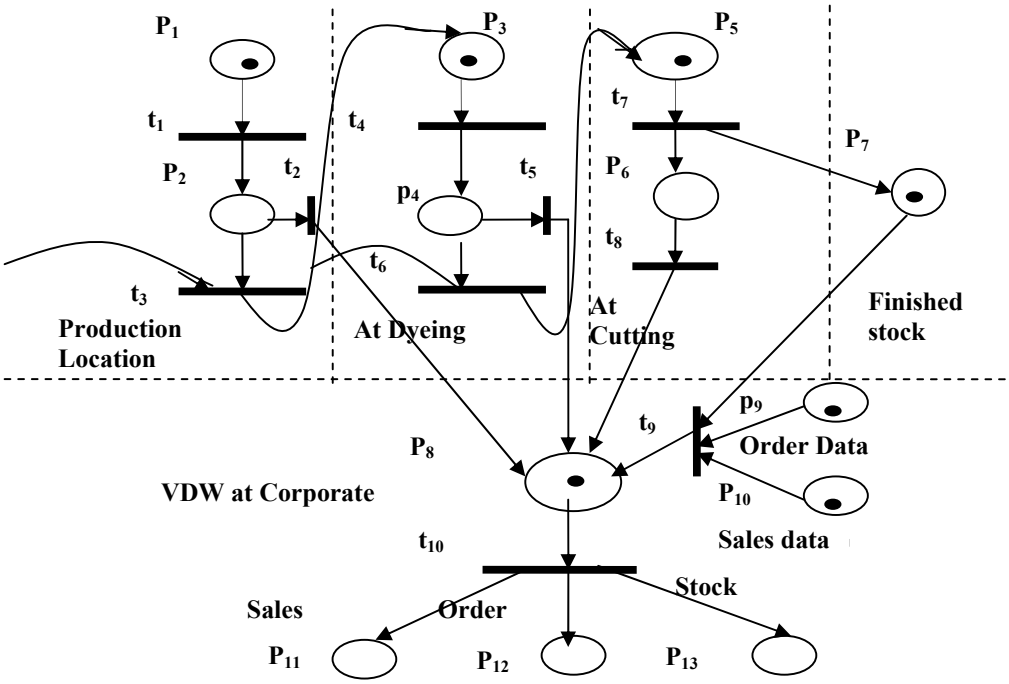

Figure 6. Petri Net model of the VDW

Table 1. Place and Transition Function

Places At Production site		Transition function	
P_1	Raw Yarn ready for Production	t_1	Knitting in process
P_2	Production Data Mart created	t_2	Knitted data move out to VDW
	At Dyeing		
P_3	(i) Produced Fabric ready for Dyeing (ii) Fabric purchased from outside source	t_3	Fabric moved to Dyeing point
P_4	Dyeing Data Mart created	t_4	Dyeing started
	At Cutting		
P_5	(i) Produced fabric ready for cutting (ii) Fabric purchased from outside source	t_5	Dyeing data move out to VDW
P_6	Cutting Data Mart created	t_6	Fabric moved out to Cutting point
P_7	Finished fabric stock	t_7	Cutting in process
P_8	Data input to VDW	t_8	Cutting data move out to VDW

P_9	Sales data from operational source	t_9	Update Stock, Sales and Order data on VDW
P_{10}	Customer Order from operational source	t_{10}	Query from VDW
	Query from Slices of VDW		
P_{11}	Stock slice Query		
P_{12}	Order slice Query		
P_{13}	Sales slice Query		

4.1.1. Incidence Matrices of the PN for the VDW

Table 2(A). Pre Incidence Matrix

Place Transition	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
P_1	1	0	0	0	0	0	0	0	0	0
P_2	0	0	0	0	0	0	0	0	0	0
P_3	0	0	0	1	0	0	0	0	0	0
P_4	0	0	0	0	0	0	0	0	0	0
P_5	0	0	0	0	0	0	1	0	0	0
P_6	0	0	0	0	0	0	0	0	0	0
P_7	0	0	0	0	0	0	0	0	1	0
P_8	0	0	0	0	0	0	0	0	0	1
P_9	0	0	0	0	0	0	0	0	1	0
P_{10}	0	0	0	0	0	0	0	0	1	0
P_{11}	0	0	0	0	0	0	0	0	0	0
P_{12}	0	0	0	0	0	0	0	0	0	0
P_{13}	0	0	0	0	0	0	0	0	0	0

Table 2(B). Post Incidence Matrix

Place Transition	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
P_1	0	0	0	0	0	0	0	0	0	0
P_2	1	0	0	0	0	0	0	0	0	0
P_3	0	0	1	0	0	0	0	0	0	0
P_4	0	0	0	1	0	0	0	0	0	0
P_5	0	0	0	0	0	1	0	0	0	0
P_6	0	0	0	0	0	0	1	0	0	0
P_7	0	0	0	0	0	0	1	0	0	0
P_8	0	1	0	0	1	0	0	1	1	0
P_9	0	0	0	0	0	0	0	0	0	0
P_{10}	0	0	0	0	0	0	0	0	0	0
P_{11}	0	0	0	0	0	0	0	0	0	1
P_{12}	0	0	0	0	0	0	0	0	0	1
P_{13}	0	0	0	0	0	0	0	0	0	1

Table 2(C). Combined Incidence Matrix

Place Transition	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
P_1	-1	0	0	0	0	0	0	0	0	0
P_2	1	0	0	0	0	0	0	0	0	0
P_3	0	0	1	-1	0	0	0	0	0	0
P_4	0	0	0	1	0	0	0	0	0	0

P_5	0	0	0	0	0	1	-1	0	0	0
P_6	0	0	0	0	0	0	1	0	0	0
P_7	0	0	0	0	0	0	1	0	-1	0
P_8	0	1	0	0	1	0	0	1	1	-1
P_9	0	0	0	0	0	0	0	0	-1	0
P_{10}	0	0	0	0	0	0	0	0	-1	0
P_{11}	0	0	0	0	0	0	0	0	0	1
P_{12}	0	0	0	0	0	0	0	0	0	1
P_{13}	0	0	0	0	0	0	0	0	0	1

The combined matrix D is computed as $[D^+ – D^-]$. The places and transitions are represented as follows: $[P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}, P_{11}, P_{12}, P_{13}]$ and $[t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}]$.
Initial marking M0 is the transpose of the row vector presented below:

P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}
1	0	1	0	1	0	1	1	1	1	0	0	0

According to equation (1), the forward, backward and combined incidence matrices are represented in table 3(a), 3(b) and 3(c).

4.1.2. Reachability Graph of the PN Model

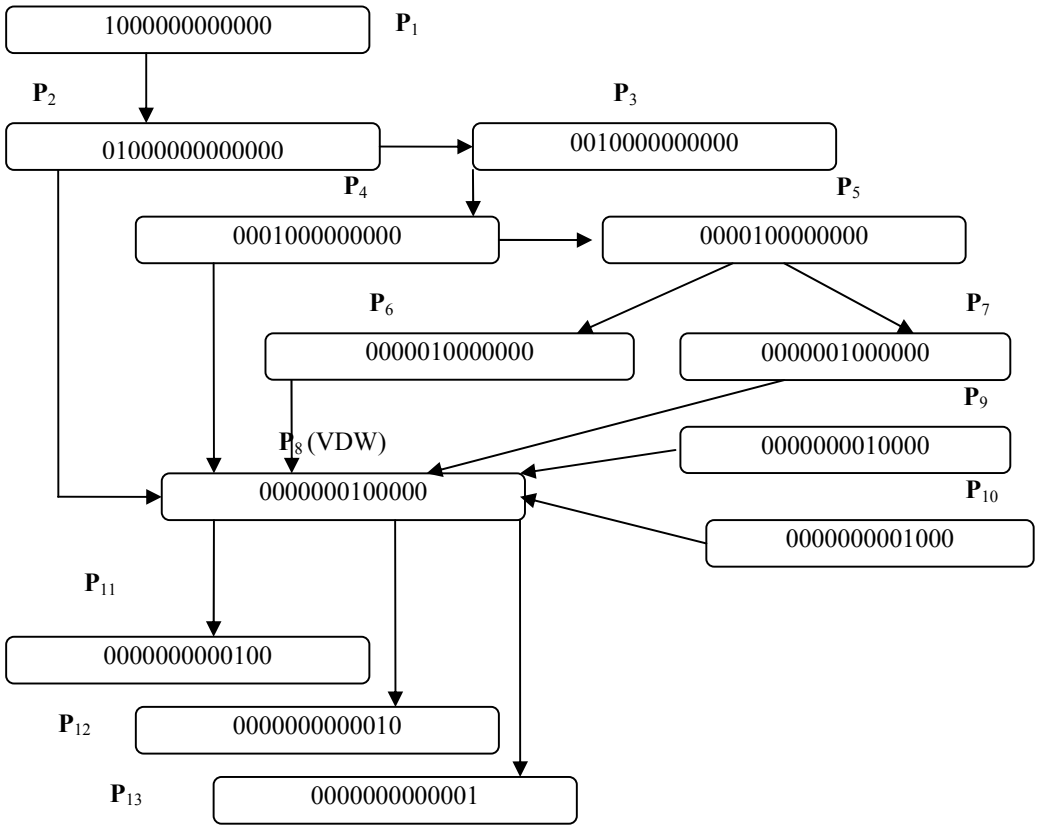

Figure 7. Reachability Graph of the (PN) Model

The Coverability or Reachability tree analysis is a dynamic enumeration process of finding all reachable markings or their coverable markings. The reachable place of a Petri Net can be expressed by the directed Reach ability graph. The nodes of the graph are identified as markings of the Petri Net $R(N, M_0)$, where M_0 is the initial marking and the arcs are represented by the transitions of N. The graph is used to define a given Petri Net N and marking M, whether M belongs to R(N). If in a Petri

Net firing of one transition does not disable the firing of another transition is a parallel activity called persistent transitions. For our speed-independent asynchronous model transitions t_1, t_4 and t_7 are parallel activities and it observes the persistent property. However, the Firing changes the token distribution in a net according to the transition rule but the equality problem is still undecided [7].

The coverabilty analysis of our Petri Net model starts from P_1, on production it reaches to P_2 the production data mart and to P8 for updation of the VDW schema. Fabric moved to P_3 for dyeing at a distant location. At P_4 the dyeing data mart is built and at P_8 updates the VDW schema coloured fabric moved to P_5 for cutting at a different location. At P_6 the finished stock data mart is update at P_8 the VDW schema. The finished fabric moved to P_7. Physical stock locations are P_7, P_9 and P_{10}; sales status and order status respectively update the VDW schema. At P_{11}, P_{12} and P_{13} the sliced queries are performed on stock, sales and order respectively. For place, transition details refer to Table 1.

4.1.3. Reachability Analysis

In order to study the behavior and the properties of the model the following analysis are performed on the Reachability Graph presented in figure 7 above [10].

Safeness : Any place of a Reachability Grapgh is declared safe, if the number of tokens at that place is either 0 or 1. For our case the graph clearly shows that any of the places [$\mathbf{P}_1$ - $\mathbf{P}_{13}$] represents a combination of 0(no token) and 1(token), which implies that if the firing occurs there will be a token at the position bit other wise no token. Thus it shows each of the places has a maximum token count 1 or 0 and is declared safe and as all the places in the net are safe, the net as a whole can be declared safe.

Boundedness: The boundedness is a generalized property of safeness. The limitation of token numbers in a place restricted to 1 in case it is safe is enhanced to some integer k, where k is known before hand for a place or we call it as a constraint to check the overflow condition at any stage calculated once at start. The boundary value for each place will be the maximum token count for that place. If there is no overflow at any place then as a whole the design guarantees the boundedness of the model. For our case at each stage from [p_1- p_{13}], k=1 and hence it is bounded.

Conservativeness: Conservation property of a Petri Net model checks the number of tokens remains constant before and after the execution. The process is to count the sum of all tokens at their initial markings. Next the Reachability tree is traversed and the sum of all tokens is calculated for each marking in the tree. For our case it is 1. If all the markings in the Reachability tree have the same sum of tokens, then the Petri Net is declared to be strictly conservative. So our model is also strictly conservative. However, it will not be out of place to mention that in most cases due to process transformation explicit token counts are difficult to obtain to prove the conservation.

Liveness: The liveness property of a Petri Net is used to show continuous operation of the net model or in other words, it can be said that the system will not get into a deadlock state. As the virtual data warehouse needs to perform resource updation, some transaction processing activity will take place. The possibility of deadlock or live lock can't be ruled out and to be checked. In order to find whether or not the Petri Net is live; move along the markings of the Reachability tree. If any marking exists in the tree such that no transitions are enabled from that marking, then that marking represents a deadlocked state, and the Petri Net lacks the liveness property. Otherwise it is declared live. For our case no such deadlock situation appears in [p_1- p_{13}]. So we call our model live.

However, the problems arise when the Reachability tree graph is used, where there exist loops in the tree. It may cause a particular place occupied with an infinite number of tokens, which results in an infinite sized tree.

5. Slicing-Based Analysis for VDW

Due to the complexity of operation and the size of the model, Reachability analysis becomes difficult. In such situations by preserving the properties of the Petri Net model reductions to a simpler model is possible. One such scalable technique is Petri Nets slicing used for Reachability analysis [9]. Let us now redefine the equation (1) of section 4 as follows:

$PN= (P, T, D^-, D^+ F, W, M)$ (2)

Where, P is the set of places and |P|=m
T is the set of transitions and |T|=n
D^-: P×T→N is the pre incidence matrix that specifies the arcs directed from places to transitions
D^+: T ×P →N is the post incidence matrix that specifies the arcs directed from transitions to places
F is a set of arcs , W is weight functions of arcs and M is the initial marking.
Let Sliced Petri Net ={P_Slice_i | i=1..n} be a set of places obtained out of slicing. where Petri Net $Slice_i = (P_i, T_i, D_i^-, D_i^+, F_i, L_i, W_i, M_i)$ satisfies the following:

- $Pi = P\_slice_i$
- $Ti = \{ t \in T \mid \forall p \in Pi, \exists(p, t) \in F \text{ or } \exists(t, p) \in F \}$
- $Fi = \{ (p, t) \in F \text{ or } (t, p) \in F \mid \forall p \in Pi, \exists t \in Ti \}$
- $Li : Ti \rightarrow \Sigma+$ is a label function of a transition, where Σ is a character set. The transition name. t_l in N to a label of t, that is, $Li(t) = t_l$,
- $W_i(p) = W(p)$, and $M_i(p) = M(p)$, $\forall p \in P_i$.

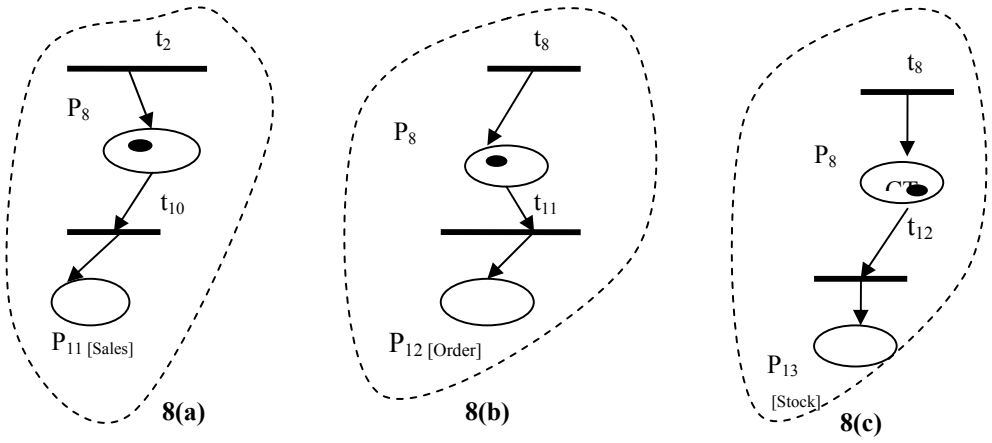

Figure 8. Petri Net slices of figure 6

Some labels may appear in multiple slices for shared transitions in a Petri Net. Let us assume that $L_i(t)$ is shared among several Petri Net slices and a transition *t* is enabled if *t* is enabled in all the slices which have the label of t. Figure 8 (a) , (b) and (c) presents the slices of the figure 6.

Here, P_8 represents the Virtual Data Warehouse in all the three slices. By enabling the transitions at t_2, t_6 and t_8 the VDW are updated from the data marts of Production, Dyeing and Cutting .respectively. The slices are abstract representations of the process that can be executed through a two dimensional SQL query instead of a complex multidimensional DMQL query form the VDW, e.g., a query on location wise sales for a particular item may be expressed as:

```
Select product, customer, date, vendor, distributor, count (*) as Qty_Sold,
sum (amount sold) as Total_sold
  from sales group by $location,product,date
    having QTY_Sold <>0;
```

Here, '$' indicates terminal inputs for variable location input. The indexing mechanism is taken care at the time of data set updation at VDW at location P_8. Similar queries on order and stock can also be executed in parallel. The slices shown here are only some samples and many such slices can be built along with PL/SQL queries for faster accessing of information compared to conventional OLAP operations on the multi-dimensional data cubes. Reachability graph of each slice can be drawn in isolation and the local properties can be analyzed. Each of the slices can be analyzed for the global properties of the system.

6. Conclusion

Interoperability between distributed systems is a common problem with present applications. Some of the architectures that are used for integration of distributed legacy systems are CORBA, Java J2EE, XML, SOAP, DCOM, Java RMI, EAI, and OMG MDA [13]. CORBA allows applications to communicate with one another efficiently and the Extensible Mark up Language XML is used to process data on the WEB. DCOM is a protocol that enables components of the architecture to communicate directly over a network. Some of the generic frame works developed for distributed decision making has been discussed in section 2, particularly in the area of internet based e-commerce applications. But for corporate and multinationals, strategic level distributed decision support system needs large volume of historical data as a backup for BI as well as transaction data for data mining and dynamic decision making .

In this paper, our focus is on building a generic model of a virtual data warehouse with the help of a Petri net model. The concept of color to mark the tokens for the parallel and concurrent processes is felt highly desirable but not used. In our future work we use the color tokens during modeling to minimize the transitions and maximize the efficiency. We also like to indicate that for large to medium sized MNC, whose business is spread across several countries and continents the analytic processing on the VDW will demand dynamic decision making at multiple sites. A centralized deployment of the integrated VDW architecture presented in section 2.1 may put a cap on the functional efficiency and the availability due to constraints in the communication infrastructure, the size of the VDW, the complexity of data updation and various common resiliencies of transaction processing.

7. References

[1] Jiawei Han, Micheline Kamber , “Data Mining concepts and Techniques” ,chapter(1 ,3 and 4), Morgan Kaufmann, Indian reprint (2004) , ISBN: 81-8147-049-4

[2] W.H. Inmon, “Building the Data Warehouse”, NY: John Wiley. 3rd ed., 2002, ISBN: 047116310-4.

[3] Ammoura Ayman, Zaiane Osmar R., Yuan Ji, “Towards Framework for the virtual Data Warehouse”, British National conf. on databases, 2001, vol. 2097, pp. 202-218, ISBN 354042265-X

[4]Jane Zaho, “Designing Distributed Data Warehouses and OLAP Systems”, Proc. of Int’l conf. on Information Systems Technology and its applications (ISTA), 2005, pp.:254-263, ISBN: 388579392-X

[5] Kaushal Chari, “Model composition in a distributed environment“, Decision Support Systems archive, Elsevier Science, Volume 35, Issue 3, (June 2003), Pages: 399-413, ISSN:0167-9236

[6] Bidyut Biman Sarkar and Nabendu Chaki, “High level Net model for analyzing agent base distributed decision support system “, Proceedings of the IACSIT, International Spring Conference, April 2009, Pages:339-346, ISBN 978-0-7695-3653-8

[7] Tado Murata, "Petri Nets : Properties, Analysis and Applications", Proceedings of the IEEE, Volume 77, No .4, April 1989, Page:541-580, ISBN: 0-387-13723

[8] Yuefeng Li, Wanzhong Yang, Yue Xu, “Multi-Tier Granule Mining for Representations of Multidimensional Association Rules”, Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM), October 2006, Page:953-958, ISBN: 0-7695-3015-X

[9] M. Liorens, J. Oliver, J. Silva, S. Tamarit and G. Vidal “Dynamic Slicing Techniques for Petri Nets”, Proceedings of the Second Workshop on Reachability Problems in Computational Models, Volume 223, December 2008, Pages 153-165,ISBN: 978-3-540-77565-2

[10] Barad, M. “Timed Petri nets as a verification tool”, Proceedings of IEEE winter Simulation Conference, 1998, Page:547-554,volume.1, ISBN: 0-7803-5133-9

[11] Pascal Wehrle, Anne Tchounikine, Maryvonne Miquel, “A Model for Distributing and Querying a Data Warehouse on a Computing Grid”, Proceedings of the 11th International Conference on Parallel and Distributed Systems , Volume 01, (2005), Pages: 203 - 209 , ISBN:1521-9097

[12] Y. Zhao, M. Wilde, I. Foster, J. Voeckler, J. Dobson, T. Jordan, E. Quigg, "Grid Middleware Services for Virtual Data Discovery, Composition, and Integration", Proceedings of the 2nd workshop on Middleware for grid computing (MGC), Vol. 76, 2004, pp.: 57 - 62 , ISBN:1-58113-950-0
[13] S. Kami Makki, and Ohio "Distributed system:An Effective information Sharing Approach for Legacy Systems", JCIT journal volume 2 (2007), pages: 22-28 (3), ISSN : 1975-9320

AUTHOR BIOGRAPHY

Nabendu Chaki is a faculty member in the Department of Computer Science & Engg., University of Calcutta, Kolkata, India. He received his Ph.D. degree from Jadavpur University, India in 2000. His areas of research interests include Distributed Computing and Software Engineering. Dr. Chaki has supervised in the Ph.D. program in Software Engineering in Naval Postgraduate School, Monterey, CA, USA during 2001–2002, as a Research Assistant Professor. He is a visiting faculty member for many Universities including the University of Ca'Foscari, Venice, Italy. Besides being in the editorial board of a few International Journals, Dr. Chaki is a KA Editor of SWEBOK Guide V3 of IEEE Computer Society. He has also served in the committees of several international conferences. His total number of publications in referred international journals and conferences is more than 70.

Bidyut Biman Sarkar is a faculty Member in MCA department of Techno India affiliated to West Bengal University of Technology. He has received his post graduate degree in Applied Mathematics from the University of Calcutta in the year 1978. Bidyut served IT industry in India, Singapore, Thailand and other South Asian countries for more than 20 years. He is pursuing his Ph.D. studies in the areas of distributed computing. A good number of publications are already to his credit in national and international conferences and journals. He has also authored a few text books for the Engineering students at the under graduate level of Computer Science and Information Technology.