International Journal of Physical Sciences Vol. 3 (11), pp. 264-274, November, 2008
Available online at http://www.academicjournals.org/IJPS
ISSN 1992 - 1950 © 2008 Academic Journals

Full Length Research Paper

A text based approach to content based information retrieval for Indian medicinal plants

Basawaraj S. Anami[1], Suvarna Nandyal[2*] and Govardhan A.[3]

[1]Department of Computer Science and Engineering, Basaveshwar Engg College, Bagalkot.
[2]Department of Computer Science and Engineering, P.D.A College of Engineering, Gulbarga.
[3]Department of Computer Science and Engineering, JNT University, Hyderabad.

Accepted 4 November, 2008

The usage of plants as foodstuff and also medicine is universal. It is important and yet a difficult task to identify the medicinal plants' species present in the universe through computers. Development of a machine vision system for medicinal plants is a dire necessity as this tacit knowledge dies with the experts in the field. Hence it is helpful to understand and manage these medicinal plants in the interest of pharmaceutical industry and preparation of home medicines. This paper deals with design and development of medicinal plants' database comprising of 500 species and an efficient text based query interface for retrieval of the desired information. The database consists of scientific and regional names, medicinal values of parts, details of each part such as leaves, flowers and the like, which act as properties or characteristics of plants. The proposed design takes care of access to these medicinal plants through the combination of multiple plants' attributes. The accuracies of recognition are reported for major and minor properties of the plants. The results have revealed that leaves and their shapes invariably help in recognition of plants. Hence, Furthermore, retrieval of relevant information through content based approaches is the ultimate target.

Key words: Database, Indian medicinal plants, query processing, image retrieval.

INTRODUCTION

Our knowledge about Indian medicinal plants in the environment is far from complete. The medicinal plants have not been cataloged over the years. This is a serious deficiency in terms of conservation, establishment of natural preserves, location and protection of unknown species. Without the knowledge of the present botanical and colloquial names of plants, it would be very difficult, if not impossible, to identify, classify and use the Indian medicinal plants. The Indians are probably one of the world's largest users of medicinal plants. Millions of rural households use medicinal plants invariably in a self-help mode with a strong belief in their ability to cure diseases. In India medicinal plants have been prescribed and used for centuries in Ayurvedic medicine. This Indian traditional knowledge of Ayurvedic medicine is gaining more popularity all over the world. The World Health Organization (WHO) reports that nearly 80% of people in the developing countries have relied on the herbal treatment. This tacit knowledge dies with the experts and difficult to carry forward it from generation to generation. This being the motivation for present work, a goal is set to design and develop a database for Indian medicinal plants.

When species and habitats disappear, in response, we lose opportunities to understand associated biological complexities of the Indian medicinal plants. New technologies and methods are required to enable rapid identifycation of medicinal species, access to bio-diverse information and construction of eco-informative knowledge. Hence, the present paper provides an information retrieval system for Indian medicinal plants, which would be highly useful to common people, farmers, ayurvedic practitioners, researchers and other agencies involved in the area of medicinal plants.

The identification and classification of medicinal plants is done based on their external features. The knowledge of plants' properties is essential to those who work in this area. Some commonly used terminology for descriptive plant taxonomy is provided in this work and is useful in understanding Ayurvedic medicine and their preparation. Designing and maintaining a systematic database to

*Corresponding author. E-mail: suvarna.mangalgi@gmail.com.

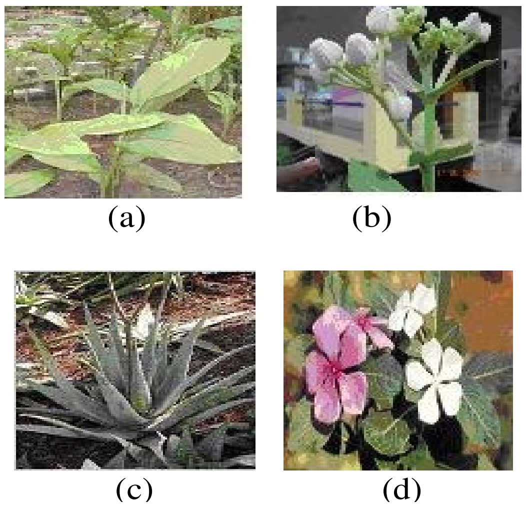

Figure 1. Samples of Indian medicinal plants; (a) Curcuma longa (b) Calotropis gigantea (c) Aloe Vera (d) Catharanthus roseus

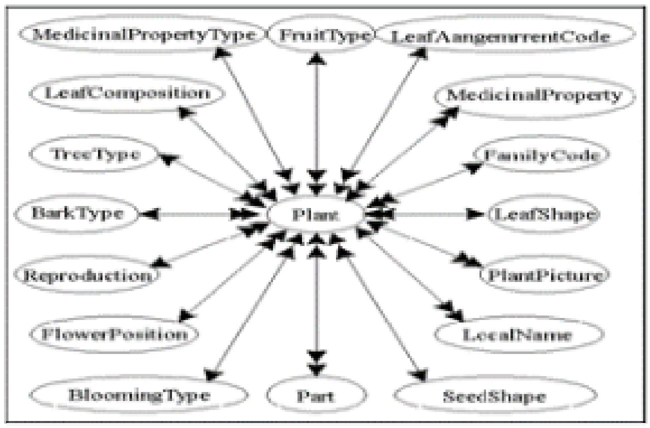

Figure 2. Medicinal plant properties

store the information about the medicinal plant is an important task. Generally, the design of databases differs from one another because of their contents. Genetic database is different from medicine database and medicine database is different from criminal record database. Hence, the design of a database for the Indian medicinal plants is quite different from the existing ones. The strength and utility of the devised database is its speed of data retrieval based on the features, namely. (i) Extracting the unique feature of a medicinal plant, (ii) Extracting the medicinal plants which posses certain properties like fruits or flowers and (iii) Extracting all or selected properties of a specific medicinal plant.

The authentic data for the developed data base is from (Warrier, 2003) and comprises of scientific name, regionnal Name, distribution, properties of the plants and the like. The property means the information about their color, height, stem fruit etc. Furthermore, as information is commonly based on human knowledge and interprettation, it is some times inaccurate and uncertain. Therefore, we need to have an automated flexible system for information storage and retrieval for medicinal plants.

Figure 1 shows some samples of medicinal plants. The major properties associated with medicinal plants are given in Figure 2.

The remaining part of the paper is organized into five sections. The review of literature is given in section 2. Section 3 deals with the proposed method. Section 4 gives experimental results and discussion. Web implementation of medicinal plant database is given in section 5.The work is concluded in section 6.

REVIEW OF LITERATURE

Over the last decade, the database research community is actively involved in varieties of data management namely, genetic data, criminal data, demographic data and the like. Some work related to plants is also cited but in the context of other countries. There is no database management work cited on Indian medicinal plants.

A database system, called BODHI (Bio-diversity Object Database arcHItecture), that is specifically designed to cater to the special needs of biodiversity applications. BODHI currently hosts purely plant-related data. The main focus of their work is to build an object-oriented database intended to process queries in multi-domain such as taxonomy characteristics, spatial distribution and genomic sequences (Srikanta et al., 2003).

A multi-faceted information on medicinal plants of India has been put together by Foundation for Revitalization of Local Health Traditions (FRLHT), Bangalore in the form of computerized databases, specialized reports, information products, websites and trade bulletins (http://www.frlht.org.in/,Encyclopedia of medicinal plants). Medicinal Plant databases, relational database system, have been described in MEDPHYT® (Kettner et al., 2005) which has data about any European plant of medicinal and pharmaceutical interest. English and German versions are available on internet. A help for protection on intellectual property Rights to the researchers and clinicians is available in (Kameshwararao, 2001). A plant conservation software called “PlantCon” is proposed (http://www.cimap.res.in, Central Institute of Medicinal and Aromatic Plants). A Flexible approach to Retrieve Plant Images using fuzzy concepts such as fuzzy subset theory and fuzzy thesauri is employed (Andres, 2000). The electronic databases available are MAPA, CAB abstracts, AGRIS, AGRICOLA, PASCAL, MEDILINE, EMBASE, APINMAP etc (Bhat, 1995).The industry and research communities developed methods for query optimization. We briefly present a few of them. The join operation and the execution of selection and projections in query optimization are given. The use of rank parameters to each operation based on selectivity and cost per tuple

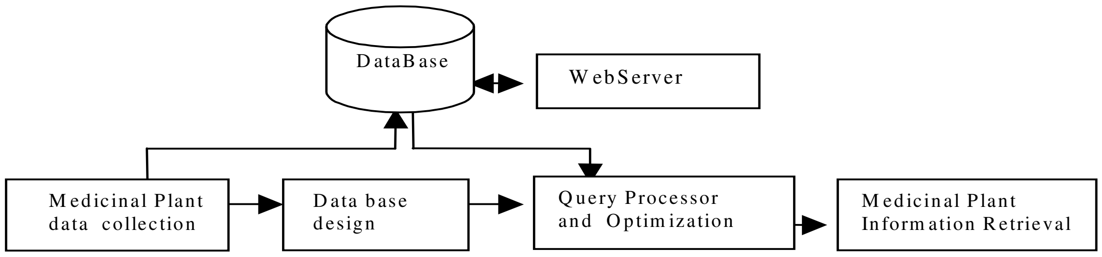
DataBase
WebServer
Medicinal Plant data collection
Data base design
Query Processor and Optimization
Medicinal Plant Information Retrieval

Figure 3. Block diagram of proposed work.

is also introduced. The heuristics in this algorithm is that lower the rank, earlier an operation is executed (Hellerstein, 1994). The performance of a database system is often determined by execution of selection, projection and order of join operations (Lee et al., 2001). A mathematical basis to (Lee et al., 2001) for representing a query and searching an execution plan is given in (Lee et al., 2001) and uses the notion of graphmodel in which node represents an operation and an edge associated with weight. The query transformation algorithms to rewrite nested SQL queries into equivalent flat queries which can be processed more efficiently is given in (Cao and Badia, 2007). Algebraic optimization rules are used by considering push down nest past join method and found that it takes shorter time with increasing sizes of relation. On the other hand, an algebra based approach has been studied (Ceri and Gottlob, 1985) and developed a translator that transforms SQL queries into relational algebra with aggregate functions. To optimize SQL queries having aggregate functions it is necessary to transform SQL queries into relational calculus and algebra (Bultingsloewen, 1987). The interference rules to support intelligent data processing in semantic query optimization is found in (Siegel et al., 1992).

To the best of knowledge of the authors, a standard Indian medicinal plants database is not available to the researchers and Practitioners. Further, it is also observed that amount of technology application effort gone into this area is very less and hence it is the motivation for the present work. We have designed database for medicinal plants keeping in mind the need for content based image retrieval of these plants. We have also proposed a technique for efficient information retrieval.

PROPOSED METHOD

The block diagram shown in Figure 3 gives important phases of the proposed method. The functions of each of the phases are detailed as follows.

The Medicinal Plant Data Collection Phase(MPDCP) involves assimilation of medicinal plant systematic description, geographic distribution, details about each part, parts used for medicinal purpose, medicinal value, botanical, vernacular names etc. There is no benchmark database available for researchers/practitioners in medicinal plant and hence, we have developed a database from (Indian Medicinal plants), which benefits researchers in this area. The proposed system is not only restricted to special scientific user groups but open to experts as well as home users interested in medicinal plants and their applications.

The Database Design Phase (DDP) contains defining tables with data about different design schema for plant properties. We have used MS Access to create databases. The Query Processor and Optimization (QPO) are devised to obtain information on plants with specific details. The QPO phase produces an efficient execution plan for processing the query which is represented by a standardized and canonical query tree. An optimizer finds an efficient execution plan for a query tree. It is observed combinatoria increase in the possible number of table join combinations as queries become more complex. An efficient query is executed by QPO to retrieve information on the medicinal plant with unique features. These individual phases are further elaborated in the following sections.

DESIGN OF DATABASE SCHEMA

Classification of medicinal plants is the process of grouping the plants together on the basis of the features, also called properties or characteristics in common. In this paper the terms features, properties and characteristics are interchangeably used. The study of plant classification is known as taxonomy. To classify plants, it is necessary to identify the features to group the plants in a logical way. Large varieties of plants give rise to diverse range of features, which need to be used for grouping. One of the oldest and commonly used methods of grouping plants is based on physical characteristics or morphological characteristics. These characteristics are shown in Table 1, which gives information about size, shape, arrangements of parts within a flower, arrangements of groups of flowers, open leaf shape, pattern of veins, stem type and shape of fruit sap color and smell of flowers etc. These characteristics are used in medicinal plant identification.

The Tables 1 - 6 show database schema devised for Indian medicinal plants. The database named PLANTS schema is given in Table 1. The different subschema FLOWERS, FRUIT_SEEDS, LEAVES, PLANT_BODY etc representing partial views of individual users of the database are given in Tables 2 - 6 respectively. PLANTS is the master database and contains the properties such as SlNo (Serial number), CollNo(Collection number) number), plant named in Hindi, Kannada, English language as Hindi name, Kan-

Table 1. Database schema plants for medicinal plants.

S/ No	Coll No	Hindi name	Kannada name	English name	Scientific name	Distribution	About_pant	Parts_ used	Property_ used
1	AVS 2358	Tikhor	Kuvehittu Tavaksiri	Arrowroot	Maranta arundina cea Linn.	Cultivated throughout India	Herb,90-180 cmhigh leaves ovate-oblong to ovate, lanceolate base rounded or cuneate, tip acute; flowers white in clusters branches, fertile stamen with appendage, ovary one-celled, one-ovuled.	Under ground rhizome	Starch of rhizome refrigerant, tonic, cough aphrodisiac, diarrhea dyspepsia, bronchitis, a nourishing food for infants, invalids conval escents. As ingredient in biscuits, cakes, puddings, jellies, face powder
≈:	:	:	:	:	:	:	:	:	≈
500	...	...	...	...	...	...	...	...	...

Table 2. Result of PLANTS database schema satisfying 1 NF with atomic values

S/No	Coll No	Hindi name	Kannada name	English name	Scientific name	Distribution	About_pant	Parts_ Used	Property_ used
1	AVS 2358	Tikhor	Kuvehittu Tavaksiri	Arrowroot	Marantaarundin acea .	throughout India	Herb,90-180 cm high	Under ground rhizome	Starch of rhizome …
1	AVS 2358	Tikhor	Kuvehitt	Arrowroot	Marant arundinacea ...	Cultivated India	leaves ovate-oblong to ovate, lanceolat	Under ground rhizome	Starch of rhizome …..
1	AVS 2358	Tikhor	Kuvehitt	Arrowroot	Marant arundinacea ...	Cultivated India	flowers white in clusters branches	Under ground rhizome	Starch of rhizome …..

nada name, English name, distribution,About_plant, Parts_used and Property used. The property About_plant includes the descript-tion of fruit, flower, leave and other properties required for the description of a particular plant. The SlNo is used as primary key. The Collection numbers are drawn from the book (Indian Medicinal plants) and are used for future upgrade and easy referral. In addition to this scientific names associated with plants are also kept as part of the database.

The Sl No. and Coll No. are used for referencing the master database PLANTS. The fields containing sub properties namely FLOWERS, LEAVES etc are allowed to have null fields against property field. The databases are subjected to normalization so as to isolate the dependencies. We have prepared a database of over 500 medicinal plants. The efficient database design determines the efficiency of information retrieval.

NORMALIZATION OF DATABASES

The PLANT database is not in 1NF because multi-valued attributes (Hwand et al., 2001), composite attributes and their combinations. Hence, the attribute such as about _plant is now decomposed to satisfy atomicity and uniqueness as shown in Table 2. But this approach produces more number of duplication, which increases redundancies. Further it introduces more number of null values if a plant does not contain flower or fruit etc. Hence to reduce the redundancies, we have formed separate sub schemas' such as FLOWERS, FRUIT_SEEDS, LEAVES, PLANT_BODY. These are self descriptive and contain atomic values as shown in Tables 3 - 6.These tables contain no grouping elements and each row has unique identifier serial number and collection number which forms concatenated key. Even though the databases can also be in 1NF format, normalizing it to 2NF removes further anomalies (Hussain et al., 2003). The subschema FLOWERS violates 2 NF, because non-prime characteristics such as color, petals, shape violates 2 NF because of FD1 and FD2. These characteristics are fully dependent on the primary key. Hence, the FLOWERS database is normalized into two subschema's FLOWERS1 and FLOWERS2 as shown in Figure 4.

In this, non-prime characteristics color and petals are associated with primary key on which they are fully dependent. Similarly other tables FRUIT_SEED, LEAVES, PLANT_BODY are also subjected to 2NF normalization.

Table 3 Decomposing plants into 1NF relations flowers

S/No	Coll No	Color	Petals	Shape	Others
1	AVS 2358	white	-	Clusters on diverging inflorescence branches	fertile stamen with appendage, ovary one-celled, one-ovuled.
≈	:	:	:	:	≈
I	...	...	...	...	...

Table 4. Decomposing plants into 1NF relations fruit_seeds

S/No	Coll No	Fruit_shape	Fruit_color	Seed_property	Overy	Seed_no
18	AVS2203	subglobose or ellipsoid berries	purplish black	-	-	2
≈	:	:	:	:	:	≈
J	...	...	...	...	...	...

Table 5. Decomposing plants into 1NF relations leaves

S/No	Coll No	Tip	Base	Hairy	Special	Shape
1	AVS 2358	Acute	base rounded or cuneate	-	-	ovate-oblong; ovate-lanceolate
≈	:	:	:	:	:	≈
K	...	...	...	...	...	...

Table 6. Decomposing plants into 1NF relations plant_body

S/ No	Coll No	Type	Height_from	Height-To	Stem	Branching	Roots
8	AVS 2516	Evergreen Tree	1000 cm	3000 cm	Smooth grey of brown band	-	-
≈	:	:	:	:	:	:	≈
L	...	...	...	...	...	...	...

FLOWERS

The design takes into account the fact that not all the characteristics are well defined for every plant. The text fields are nullable other than the plant name or the collection number. Hence these two fields become unique and therefore considered as the primary key. There is no necessity for 3NF because the database does not have transitive dependencies among the characteristics of database sub schemas' namely, Leaves, Flowers etc. We have allowed the text fields to be nullable so as to make the data base automatically accommodate further characteristics.

The normalization is tested based on join dependency. It is important to test for retrieval of any spurious data or null rows once the databases are joined and a query is selected. All the sub-databases possess the secondary characteristics and contain the collection number and the serial number and hence multiple values cannot be retrieved. Thus, the proposed design of database satisfies 2NF. Whenever queries are submitted to this normalized database, efficient retrieval of information with least access time and least space utilization are possible.

EFFICIENT QUERRY PROCESSING

A typical query processing on a Medicinal Plant Database for retrieving the required information is shown in Figure 5. The typical stages through which a query proceeds have the following functionality.

The Query Parser checks for the validity of the query syntax and syntactically correct query is translated into an internal form. We have used a query tree, which is an useful representation for relational calculus expressions. The Query standardizer examines all the algebraic expressions that are equivalent to the given query and chooses the one that has the least cost. By cost we mean the number of tuples executed. The Code Generator transforms the access plan generated by the standardizer into calls to the query processor. The Query Processor is responsible for actual execution of the query.

Queries may have alternate execution plans, which are equivalent in terms of their final results, but vary in their costs and expressed as the amount of time needed to execute a query. There are two

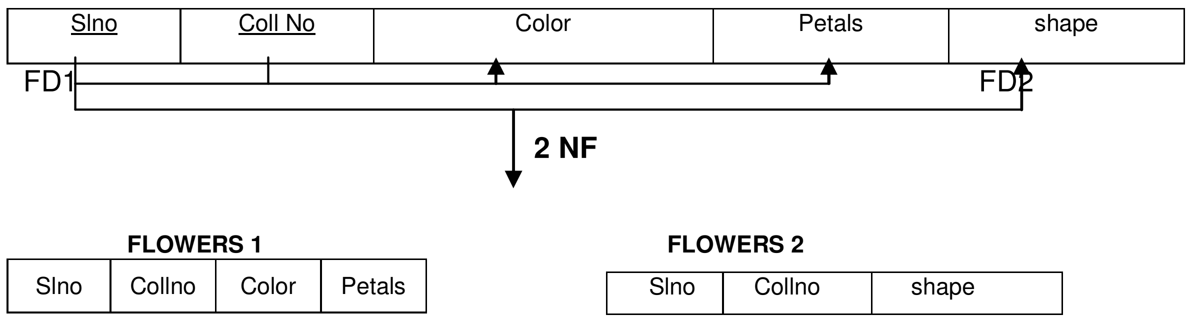

Figure 4. Normalizing flowers sub schema into 2NF relations.

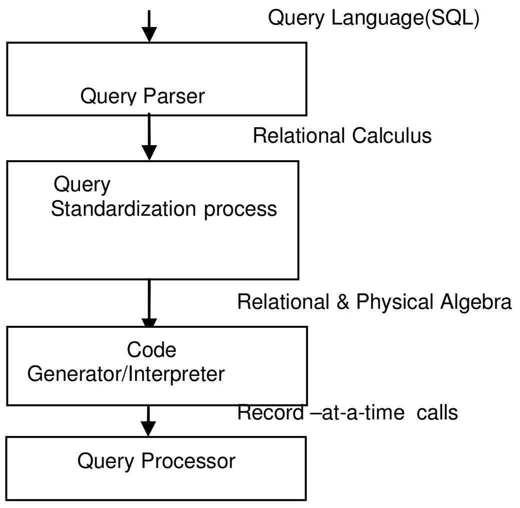

Figure 5. Query flow through a medicinal plant database.

Table 7. Plants database parameters

Relation	Cardinality
PLANTS	100
FLOWERS	84
LEAVES	100
FRUIT_SEED	80
PLANTBODY	100

ways for making a query efficient namely, Heuristic approach and Cost Based approach.

In cost-based approach, specific information about the stored data is used. This information is extremely system-dependent and includes information such as file size, file structure types, available primary and secondary indices, and the attribute selectivity. Since a cost-based approach uses the knowledge of the underlying data and storage structures, it is not considered in this work. The goal of any approach is to retrieve the required information as efficiently as possible, we have used heuristic approach.

HEURISTIC APPROACH

Heuristic approach is a rule-based method for producing an efficient query execution plan. A structured query (SQL) is first converted into standard relational algebraic expression using the operators namely, Selections (denoted by σ), projections (denoted by ∏) and joins (denoted by ∞). Typically, such an algebraic query is represented by a *Query Tree,* whose leaves are database relations and non_leaf nodes are algebraic operators. Thus, the edges of a tree represent dataflow from bottom to top.

Once a query tree is constructed, the heuristics is applied to transform a given query into a more efficient representation. Using relational algebraic equivalence rules, we have ensured that no necessary information is lost during the transformation of the tree. The heuristic rules help to break the conjunctive selects into cascading selects and to move selects down the query tree to reduce the number of returned "tuples." The projects are moved down the query tree to eliminate the return of unnecessary plants' characteristics. A Cartesian product operation followed by a select operation is combined into a single join operation. Using these steps we have observed that the efficiency of a query is improved and it is further enhanced by rearranging the remaining select and join operations so as to reduce the amount of system overhead.

We have found out from the experimentation that the queries based on certain selection criteria, the execution orders fetch minimum and maximum number of rows. The heuristic rule is said to first execute the queries with minimum result and further and them with the other queries, if required. This is the reason why we have kept the specialization query as the inner most query. Even though most of the database design works tend to represent the queries in a mathematical form, we have followed rather accurate query illustration for ease of understanding and fluidity in the representation. The goal of optimization in this work aims at minimizing the number of joins and hence we have considered the exact query and the query tree. Consider the query "Retrieve the medicinal plants with ovate leaves". The inner query is written as under

$$\sigma_{\text{shape = 'ovate'}}\text{(LEAVES)}$$

From the experiment, we observed that this query returns minimum result of around 30% from the desired database. Hence, we have identified the plants whose leaves are ovate, identification of their unique features becomes easier. From the analysis of result, we have found that the number of tuples returned for the characteristics 'root' is minimum and for 'leaves' it is maximum, as evident from Figure 9. Based on this fact, we have achieved an efficient execution plan, whenever a query is applied. We have tested queries on this developed medicinal plants' database containing 100 plants. Tables 7 and Table 8 gives relation size and selectivity

Table 8. Selectivity and dilation factor for query with ovate leaves.

Tree/Condition	Selectivity factor	Dilation factor
T1 / C ovate	0.29	1.2
T2 / Cjoin	1.0	1.2
T3/ C ovate with push down selection and projection	0.29	0.85

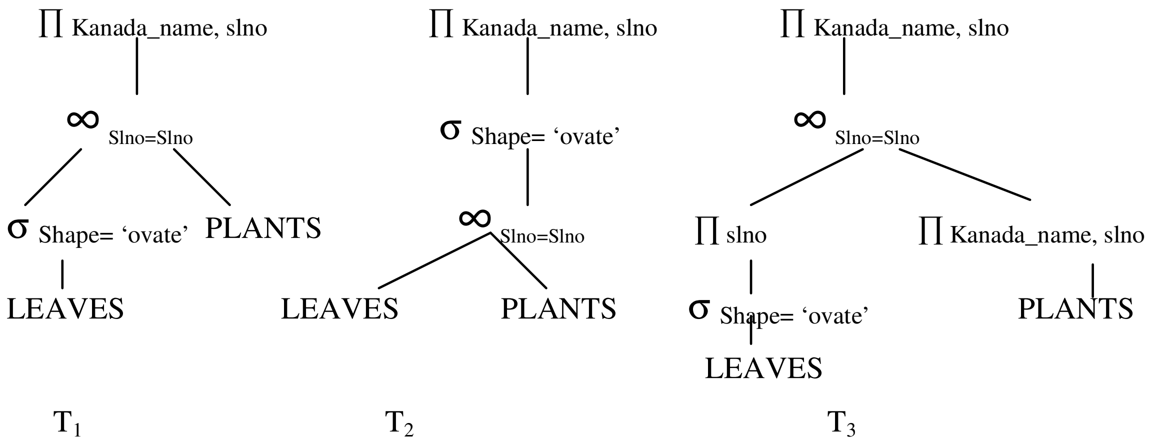

Figure 6. (i) T_1 and T_2 are Query Trees (ii) T_3: Efficient Tree.

and dilation factor for the query, *Retrieve the plants with Kanada name and SlNo and having ovate leaves*. From our database, we have found that each tuple for plants database has a size of 200 bytes and for Leaves 150 bytes. The query is written in relational algebra as follows.

$\prod_{\text{Kanada\_name, slno}} ((\sigma_{\text{shape= 'ovate'}}(\text{LEAVES})) \infty_{\text{LEAVES. slno = PLANTS. slno}}$ PLANT)

The possible three query trees for the above query are shown in Figure 6. The performance of this database system is often determined by the execution order of select-project-join operation. From literature survey, we have found out that almost all the past methods for query effectiveness adopt an existing algorithm or a modified form to find a effective solution for a given large search space. Some techniques using combinatorial algorithms (Swami and Gupta, 1988), (Swami, 1989), such as iterative improvement (Ioannidis and Kang, 1990) and simulated annealing (Kirkpatrick, 1983; Ioannidis and Wong, 1987), require a long execution time. A major problem of these methods is their lack of formal representtation to operate on query parameters, such as relation size and join selectivity factor. Hence, we have compared the execution cost in terms of space. Lee et al. (2001) proposed that "PushDown" algorithm is the best algorithm compared other algorithms proposed in (Hellerstein, 1994). The PushDown algorithm always tries to push the selections and projections down a query tree and joins are always need to be arranged in such an order that a join with a smaller selectivity factor is performed first. Using this concept, we have computed the statistical information for a query tree and generate an effective execution plan of the corresponding query tree. By effective we mean the process that uses least number of tuples.

We have used Selectivity Factor (SF) and Dilation Factor (DF) for the binary and unary operations as a good measure for space computation. We have formally defined the following parameters that are used in space representation.

Selectivity factor (SF): Selectivity factor is a factor representing the ratio of the cardinality of the result to the cardinality of the input relations. For the binary operation (join) and unary operations (selection and projection) the SF of given predicate *pi* is defined as:

$$SFi = \begin{cases} |\,Resi\,| \;/\; |R|\times|S| & \text{if } pi \text{ is a binary operation} \\ |\,Resi\,| \;/\; |R| & \text{if } pi \text{ is a unary operation} \end{cases}$$

Where |X| means the cardinality of relation X. and where *Resi* stands for the result of predicate *pi* and *R* and *S* are input operands of *pi*.

Dilation factor (DF): Dilation factor is a ratio of the result tuple size (width) to the sum of the base relations tuple sizes. Formally, it is defined as follows:

$$DFi = \begin{cases} ||Resi|| \;/\; ||R|| + ||S|| & \text{if } pi \text{ is a binary operation} \\ ||Resi|| \;/\; ||R|| & \text{if } pi \text{ is a unary operation.} \end{cases}$$

Where ||X|| means the width of each tuple X.

We have used these two factors to derive the query optimization process. We have considered a good query processing strategy represented by 'T3'compared to strategies represented by T1 and T2. In T1 selection is done at intermediate node and this returns tuples with more cardinality and size during join operation. The selectivity factor is more. From our experiment we have found out 29 plants with ovate leaves are retrieved. SFi = 29/100 = 0.29. All 29 plants with all attributes (around 5800 Kb) having more dilation factor is involved in join process. This join operation is more expensive. In tree T2, first plants which have got leaves property are joined. In case of any plant whose leaves property is not defined then that plant is not considered for join operation which gives less SFi. In T2, selection is made after join, which generates unnecessary properties and leads to increase in size and gives SFi

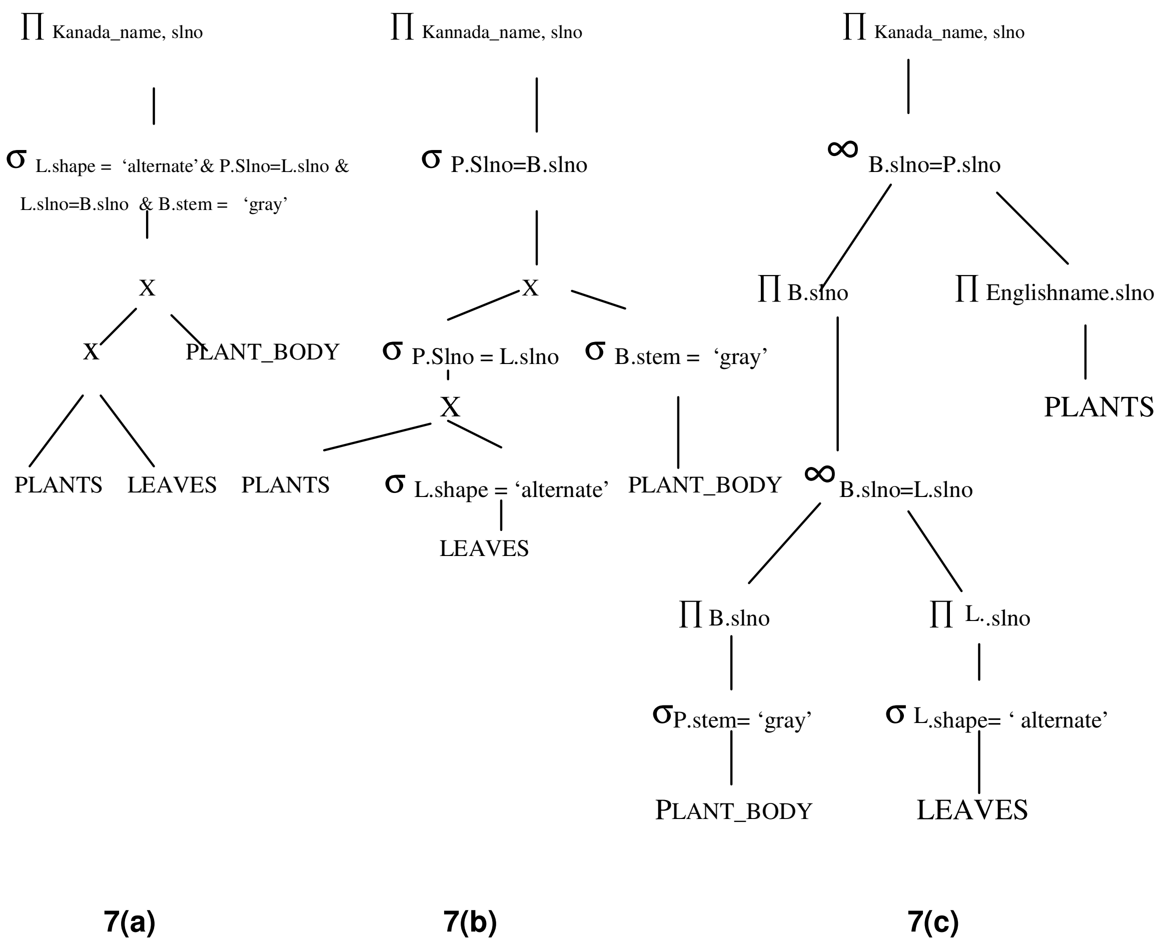

Figure 7. Execution plan for complex query (a) Initial Canonical tree. (b)Moving select operations down the tree (c) Applying the more restrictive select operation first , replace Cartesian product and Select with join operation.

100/100 = 1. As per experimental results, Figure 9, we have found leaves are most prominent parts for recognizing the plants. All plants are defined with leaves. Hence, this provides an expensive execution plan.

In case of tree T3 it is effective because the selection and projection are moved down the tree. So that it reduces the number of intermediate properties and cardinality and SFi as shown in Table 3. The DFi during join operation at intermediate level is 0.85 and SFi is 0.29. Hence, we have inferred that the performance of push-down strategy gives an effective execution plan. Hence, it is necessary to select the properties which lead to smaller number of tuples with less size and storage. Such operations are evaluated first. Since the execution plan is dictated by the database design, the queries are rendered effective. This is illustrated by a fairly complex query on the database and is shown in Figure 7. Consider a query corresponding to Retrieval of Kanada Names, SlNo of plants with alternate leaves and gray color stem.

```
SELECT P.Slno,P.Kanada_name
 FROM PLANT as P
 WHERE P. slno IN ((SELECT L.slno
                    FROM LEAVES as L
                   WHERE L.shape= 'alternate' and L.slno IN
                    (SELECT B.slno
                    FROM PLANT_BODY as B
 WHERE B.stem= 'gray' AND B.slno = L.slno))
```

The query is represented in relational algebra as follows.

LEAF_PROP $\leftarrow$ σ shape= 'alternate'(LEAVES)

STEM_PROP $\leftarrow$ σ stem= 'gray'(PLANT_BODY)

PLANT_LF_STEM $\leftarrow$ LEAF_PROP ∞ LEAF_PROP.slno = STEM_PROP.slno STEM_PROP

UNIQUE_PLANT $\leftarrow$ $\prod$Kanada_name, slno (PLANT_LF_STEM ∞ PLANT_LEAF_STEM.slmo= PLANTS.slno PLANTS) From the experiment we have found that eight plants with alternative leaves are retrieved Then for the query with gray stem, five plants are retrieved. If we apply the join operation among these two operations, we have got only one plant with kanada name 'kanagi' having both alternative leaves and gray stem retrieved. Hence, these properties become unique features for recognizing plants. The transformation rules are applied and alternative execution plans are generated as in Figure 7(a) - Figure 7(c). From the database, we have obtained only few plants with stem than leaves. We have applied the heuristic rule for selection of stem relation followed by leaves. The efficient query execution plan is shown in Figure 7.

The semantics of the query is captured in initial query tree from Figure 7(a). Executing the initial query tree directly gives a very large Cartesian product of all PLANTS, LEAVES, PLANT_BODY. By applying the heuristics rules (Frasincar et al., 2001) improvement is observed by switching the positions of relations and select operation. All three query trees are equivalent. To understand why it is more efficient to execute the query tree of Figure 7(c) instead of either the tree of Figure 7(a) or the tree of Figure 7(b), we have given here a quantitative information for our database containing information on 100 medicinal plants. For the tree shown in Figure 7(a), we need to compute the Cartesian product 100 x 100 x 100 = 10^6 elements. For the tree shown in Figure 7(b), we require 8 x 100(plants with alternate leaves) + 5 x 100 (plant with gray stem) = 1200 elements and for the tree shown in Figure 7(c), we require 5 x 8 = 40 elements. Of these, only one (plant with both alternate leaves and gray stem) plant is retrieved. It is observed that the tree in Figure 7(c), is most effective.

The effectiveness of the database design is determined by the number of iterations or the attempts taken to define a unique properties of the plants. We have considered 10 major properties, all together, hence the maximum number of possibilities is 10 for any given plant. In large databases, this is considered quite a small number of iterations. Hence, we have inferred that the proposed design gives good results for any plant.

EXPERIMENTATION

The characteristics differ from plant to plant, we have designed a view-based approach that takes input parameters from the user and passes them to the query processor to generate the results. We have created the relations in the view level. Many entries in data-base PLANT is null as their values are not well defined or do not exist. It not only suppresses the insert or update or delete ano-malies but also minimizes the normalization overhead. The main purpose is to uniquely identify the plants, immaterial of normali-zation of the database for optimum processing and storage. Some of the examples of common queries that are supported are given as under.

(i) Classification based on the base tip of leaves

```
SELECT[plants].[kanada_name],[plants].[hindi_name],[plants].[sci_
name],[plants].[distribution],[plants].[parts_used],
[plants].[about_plant]
FROM plants, leaves
WHERE([plants].[slno]=[leaves].[slno]) And
([plants].[collno]=[leaves].[collno]) And (([leaves].[base] Like
[@base]) Or ([leaves].[tip] Like [@tip]));
```

(ii) Classification based on flower color

```
SELECT[plants].[kanada_name],[plants].[hindi_name],[plants].[sci_
name],[plants].[distribution],[plants].[parts_used],[plants].[about_pla
nt], [flowers].[color], [flowers].[shape]
FROM plants, flowers
WHERE(([plants].[slno]=[flowers].[slno])And[plants].[collno]=[flowers
].[collno]) And (([flowers].[shape] Like [@shape]) Or
([flowers].[color]=[@color])))
ORDER BY [COLOR];
```

(iii) Classification based on fruit shape description

```
SELECT *
FROM fruit_seed
WHERE len(fruit_shape)>1;
```

(iv) Classification based on fruit seed

```
SELECT[plants].[kanada_name],[plants].[hindi_name],[plants].[sci_
name],[plants].[distribution], [plants].[parts_used],
[plants].[about_plant]
FROM plants, fruit_seed
WHERE(([plants].[slno]=[fruit_seed].[slno])And([plants].[collno]=[frui
t_seed].[collno])And(([fruit_seed].[fruit_shape]Like[@Fruit_shape])O
r [fruit_seed].[seed_property]=[@Seed])));
```

The variable names (preceded by @) are used to pass values because the statements are all compounded statements and the '*value*' format detects the key words even when they are embedded in a sentence along with other words. Since "height" attribute is numeric, absolute values are be passed in order to detect the plants with specific heights. The aggregate function called *count* is used for counting the number of rows.

RESULTS AND DISCUSSION

The querying allows the user to do a complex and flexible search, especially when the database is large. In this work, user expresses a detailed query, using conjunct-tions and disjunctions. Moreover, in the case of medicinal plants, we use continuous property, such as the leaf size, or an interval data such as between 2 cm and 5 cm or a linguistic variable such as small. It is also common to use modifiers such as very small to attribute a different level of importance. An illustration of global and fragmentation schemas for medicinal plant database for retrieval of the Plants English name with white flower, conical fruits and herbaceous is given below.

Global schema

PLANTS(slno,collno,hindiname,kanadaname,englishname,scietificname,distribution,about_plant,parts_used,property_used)
LEAVES(slno,tip,base,spacial,hairy, shape)
FRUIT(slno,fruit_shape,fruit_color,seed_property,ovary,seed_no)
FLOWER(slno,color,shape,petals,others)
PLANTBODY(slno, type,height_from,height_to,stem,branching,roots)

Fragmentation Schema

PLANT_FL = SL $_{\text{FLOWER.color= 'white'}}$ PJslno (FLOWER)
PLANT_FR= SL $_{\text{FRUIT.shape= 'conical'}}$ PJ slno (FRUIT)
PLANT_TP= SL $_{\text{PLANTBODY.type= ' herb'}}$ PJ slno (PLANTBODY)
PLANT_TEMP = (PJ $_{\text{slno}}$ (PLANT_FL JN $_{\text{PLANT\_FL.slno = PLANT\_FR.slno}}$ PLANT_FR))
PLANT_RECN = PLANT_TEMP JN $_{\text{PLANT\_TEMP.slno = PLANT\_TP.slno}}$ PLANT_TP
PLANTNAME_RECN_UNIQUE=PJ$_{\text{slno,English name}}$(PLANTS JN$_{\text{(PLANTS.slno = PLANT\_RECN.slno)}}$ PLANT_RECN)
Plants_with_white_Flower={2,3,4,63,71,72,75,10,83,54,84,62,8,33,55,16,1,7,9,13,18,34,5,11,6,20,61}
Plants_ With_Conical_Fruits={7,54}
Herb_Plants={1,2,4,5,6,12,19,25,27,32,33,36,37,42,45,49,54,55,60,65,66,74,75,76,77,82,84}
White flower$\cap$ Conical_Fruits $\cap$ Herb_Plants={54}
Therefore plant 54 is a herb which possesses a conical

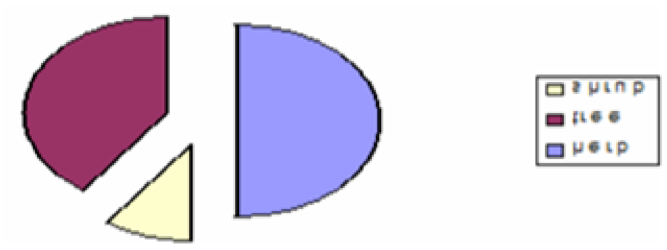

Figure 8. Percentage of different Medicinal Plants

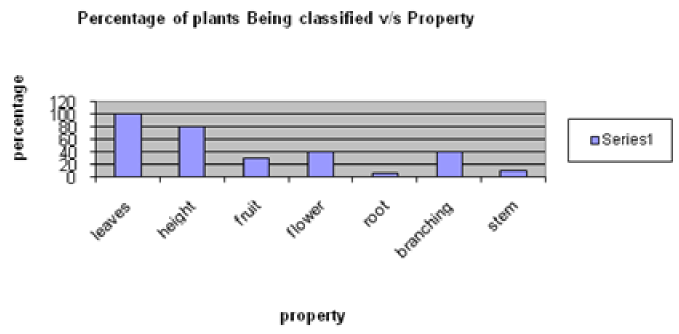

Figure 9. Classification of medicinal plants based on the plant properties

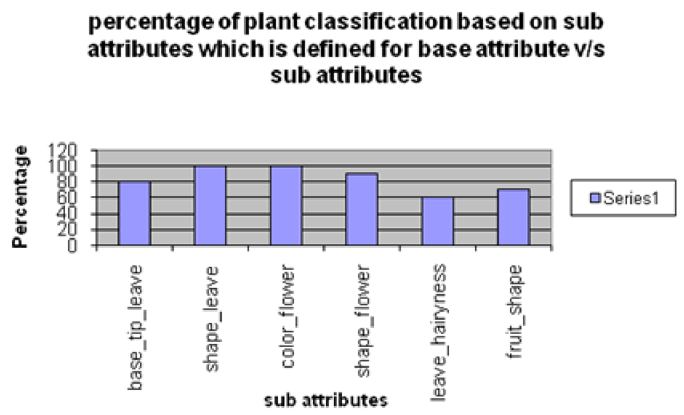

Figure 10. Classification based on the plant sub properties.

fruit with white flower
Consider query for retrieval of the plants with white flower and yellow fruit
Plants_Fruit_Color_Yellow={3,21,28}
Hence Plants_With_White_Flower $\cap$ Plants_Fruit_Color_Yellow=3

Hence, a plant with serial number 3 has white flower and yellow fruit which is unique feature for recognizing the plant. We have presented some preliminary results of experimentation on the designed database. A large number of queries are executed on the database, where plants are divided mainly into three categories herbs, shrubs and trees. Figure 8 gives the percentage of retrieval of different medicinal plants from the database. The number of database entries items from which the plant needs to be identified is brought down to nearly 50% for most of the time and nearly 10% for shrubs.
Hence, the conjugate queries usually require plant classification based on plant type. Shrubs are in small numbers and considered as the first query in the whole optimization process.

From the graph shown in Figure 9, we conclude that leaves, height, flower, branching and fruits are the most common properties, which are defined for majority of the plants and amongst them leaves are the most well defined properties in the recognition of plants. Hence the retrieval percentage is high.

The Plants, with properties given in Figure 9 constitute 90% of the entire database. There are some plants, which have the same global attributes for two different plants. Hence, in order to classify, we have found out some minute local properties specific to a particular plant. Such attributes are used to differentiate between set of similar plants. The Figure 10 gives the graph of percentage of retrieval versus the sub properties. The plants are segregated based on the sub properties of each property combined in a single query. From the experimentation, it shows that most of the plants are well defined for these common properties.

MEDICINAL PLANT DATABASE ON WEB

In order to keep the database for online use, it is web enabled and implemented using ASP.Net and Web Matrix Server. This representation of schema shows, how easily the plant attributes are fragmented and analyzed. The OLEDB database interface is considered, which involves the MS-Access database access independent of (dot) Net core component ADO.Net, integrated with Web Matrix. It resolves the connection string as the direct path to MS-Access, which is located in the web directory. For record representation, a serialized object (DataGrid) and one non serializable object (DataReader) are used. The Grid is used for displaying and joining results, where the Reader is used to fetch independent records.

Conclusion

The medicinal plant database creation and content based information retrieval suggested in this paper are useful to researchers and practitioners in this area. It is an uni-que effort to design a web enabled database on Indian medicinal plants. Future work focuses on combining text and image for content–based information retrieval. The proposed methodology has given 80% accuracy for a unique property. Hence, entire database is made of completely classifiable with defined values for the sub properties. Ultimate goal is to develop a machine vision system for medicinal plants considering properties and sub properties for their retrieval.

REFERENCES

Andres F (2000). A Flexible approach to Retrieve Medicinal Plant Images, National Institute of Informatics Journal, 12(1): 23-31.

Bhat KKS (1995). Medicinal Plants information databases, Global Initiative For Traditional System of Health(GIFTS), UK.

Bultingsloewen GV (1987).Translating and optimizing SQL queries having aggregates ,proceedings of the International Conference on Very Large Databases(VLDB), pp. 235-243.

Cao B, Badia A (August 2007). SQL Query Optimization through Nested Relational Algebra", ACM Transactions on Database Systems University of Louisville, 32(3):1-46

Ceri S, Gottlob G (1985). Translating SQL into Relational algebra: Optimization,semantics and equivalence of SQL queries, IEEE Transactions on Software Engineering, 11(4): 324-345.

Chaudhuri S (1998). An Overview of Query Optimization in Relational Systems", ACM , proceedings of Symposium on Principles of Database Systems (PODS), Seattle, USA, pp. 34-43.

Frasincar F, Houben G, Pau C (2001). XAL: an Algebra for XML Query Optimization, proceedings of Thirteenth Australasian Database Conference ADC 2002, Melbourne, Australia, pp. 49-56.

Hellerstein J (1994). Practical predicate placement, Proceedings of ACM International Conference on Management of Data (SIGMOD), held at Minneapolis, USA, pp. 325-335.

http://www.cimap.res.in, Central Institute of Medicinal and Aromatic Plants, Lucknow, India.

http://www.frlht.org.in/,Encyclopedia of medicinal plants .

Hussain T, Shafay S, Mian MA (2003). Eliminating Process of Normalization In Relational Database Design" , Proceedings IEEE International Multitopic Conference (INMIC), pp. 408- 413.

Ioannidis Y (1996). "Query Optimization",ACM Computing Surveys, symposium issue on the 50th Anniversary of ACM, 28(1): 121-123.

Ioannidis YE, Kang Y (1990). Randomized Algorithm for Optimizing Large Join Queries, Proceedings of ACM Special Interest Group on Management of Data (SIGMOD) International Conference , pp 312-321.

Ioannidis YE, Wong E (1987). Query Optimization by Simulation Annealing, Proceedings of ACM Special Interest Group on Management of Data (SIGMOD) International Conference , pp 9 - 22.

Ji_Xiang Du, Xia_Feng Wang, Guo_Jun Zhang (2007). Leaf shape based Plant Species Recognition, Applied Mathematics and Computation, 185: 883-893.

Joseph M, Hellerstein J (June 1998). Optimization Techniques for Queries with Expensive Methods, ACM Transactions on Database Systems, 23(2): 113-157.

Kameshwararao C (2001). Database Of Medicinal Plants, Current Science Karnataka state council for science and Technology , Govt of Karnataka state, India, 80(3): 463-464

Kettner C, Kosch H, Lang M, Lachner J, Oborny D and Teppan E (2005). Creating a Medicinal Plant Database Proceedings of the Workshop on Database Issues in Biological Databases (DBiBD) in conjunction with the Tenth International Conference on Database Theory (ICDT), Edinburgh, Scotland.

Kirkpatrick S, Gelatt CD, Vecchi MP (1983). Optimization by Simulating Annealing,Science New Series, 220(4598) : 671-680.

Lee C, Shih C, Chen Y (2001). A Graph-theoretic model for optimizing queries involving methods, The Very Large DataBases (VLDB) Journal, 9: 327-343.

Lee C, Shih C, Chen Y (2001).Optimizing Large Join Queries using a Graph-based approach, IEEE Transaction on Knowledge and Data Engineering, 13(2): 298-314.

Lee D, Barber R, Niblack W (1994). Indexing for Complex Queries on a Query- By- Content Image Database, Proceedings of 12th International Conference on Computer Vision and Image Processing, 1(9-13): 142-146

Mark I, Becker HJD, Lin J (2001). Representing multivalued attributes in database design, proceedings of International Association and Computer Information Systems (ICIAS), pp. 160-166.

Rajasri Bhattacharyya, Sabita Bhattacharya and sidhartha chaudhuri,(2005). Conservation and documentation of the medicinal Plant resources of India journal, Springer Netherlands publication, 15(8): 2705-2717.

Siegel M, Sciore E, Sharon Salveter (1992). A Method for Automatic Rule Derivation to Support Semantic Query Optimization, ACM Transactions on Database Systems, 17(4): 563-600.

Srikanta JB, Haritsa JR, Sen SU (2003).The Building of BODHI, a Bio-diversity Database System, Information Systems, Vol. 28, No 4.

Srikanta JB, Jayant RH, Sen SU (2004). BODHI: a DATABASE habitat for Biodiversity information, ACM International conference on Special Interest Group on Management of Data(SIGMOD), held in Paris, France.

Swami A (1989). Optimization In Large Join Queries: Combining Heuristics and Combinatiorial Techniques, Proceedings of IEEE International Conference on Data Engineering, pp. 367-376.

Swami A, Gupta A (1988). Optimization of Large Join Queries, Proceedings of ACM International conference on Special Interest Group on Management of Data (SIGMOD) International Conference, pp. 8-17.

Warrier PK, Nambiar VPK, Ramankutty C (2003). Indian Medicinal plants –A Compandium of 500 Species, Published by Orient Longman PVT Ltd, Hyderabad, INDIA , 3rd Edition, Vol. I – V.