AN INTRODUCTION TO DEDUCTIVE DATABASE AND ITS QUERY EVALUATION

Ashish Kumar Jha, College of Vocational Studies, University of Delhi, Sushil Malik, Ramjas College, University of Delhi (corresponding author)

Abstract

A deductive database consists of rules and facts through a declarative language such as PROLOG/DATALOG in which we specify what to achieve rather than how to achieve.

Facts are specified in a manner similar to the way relations are specified but it does not include attribute names whereas rules are virtual relations (views in a relational database) that are not actually stored but that can be formed from the facts by applying inference mechanism. Rules can be recursive that cannot be defined in the terms of basic relational views.

There is an inference engine within the language that can deduce new facts from the databases by interpreting these rules. The model used for deductive database is closely related to the relational data model and mainly to the domain relational calculus formalism. This deductive databases approach is also related to the field of logic programming and the prolog language.

Keywords

Fact, Rules, Clause, Prolog, Datalog, Predicate.

Introduction

For more than a decade after the development of first widely used database system, artificial intelligence and database research followed different paths. AI researcher worked on topics such as knowledge representation, natural language processing and on the other hand database researcher concentrated on the efficient storage and retrieval of information in secondary memory as well as concurrency and data security.

But there is a convergence of artificial intelligence and database system in deductive database. In deductive database system there are mainly two components: database storage and the logic to derive information.

A deductive database is a database system that performs deductions. This implies a deductive database can conclude additional facts based on rules and facts stored in the database itself. In recent years deductive database such as DATALOG and PROLOG have found new applications in data integration, information extracting, security, cloud computing and program analysis using its powerful inference engine. This research paper provides an overview of some deductive database theories and query evaluation based on given facts and rules and these are programmed in DATALOG/PROLOG. A DATALOG/PROLOG is a language typically used to specify facts, rules and queries in deductive database. The PROLOG/DATALOG search engine processes these facts and rules to give the answer of the requested query.

The meaning of terms and formulas may be defined by means of interpretation. An interpretation consists of a domain of objects say D and an interpretation of non-logical symbols in D such that a constant symbol is interpreted as an element of D and a predicate symbol is interpreted as a relation on D.

The middle part of this research paper contains various explanations of the concepts and results related to the Horn clauses. These concepts and results are accepted to a very large extent and can be incorporated easily in many working systems in the area of Artificial Intelligence.

Main Contents

A deductive database may be defined as a triple DB=(C,P,I) where C is a finite set of non-logical symbols(constants and predicates) that define a specific language. It is assumed that C has at least one constant symbol and at least one predicate symbol.

P consists of a finite set of axioms in the language and may contain rules. I is the finite set of sentences in the language, the integrity constraints that must be satisfied by the database. Update to the database typically involve P only, the updated database must still satisfy I.
Notations used in PROLOG or DATALOG are based on predicates. A predicate is a name with arguments just like a function name in programming languages. If arguments are all constant values, it simply states that a certain fact is true. A Predicate establishes a relationship among objects (arguments supplied are called objects).

A DATALOG is a variation of prolog in which a program is built from basic objects called atomic formula. In DATALOG, atomic formula are literals of the form P(b₁,b₂,b₃,………bₙ) where P is the name of predicate and b₁,b₂,b₃,………bₙ are arguments of P

PROLOG/DATALOG consists of a number of built in operators which are used for constructing atomic formulas. For example <,<=,>,>=,/= etc

Evaluation process

Phase I: Storage of rules and facts

In DATALOG formulas are first converted into clausal form before they are expressed in Datalog and these clausal forms are called Horn clauses. As it is clear that a formulas may contain quantifiers existential(for some) and universal (for all), clausal form of a formula is constructor in the follow was :

- Quantifiers are removed
- The resulting formula is made up of a number of clauses.

Where each clause is composed of a number of literals connected by OR logical connection or AND (conjunction) logical connection.

A literal can be positive or negative for example

NOT(A₁) OR NOT (A₂) OR ------- OR NOT (Aᵣ) OR B₁ OR B₂ OR ------- OR Bₙ

This clause has n negative literals (which are within NOT) and n positive literals (without NOT).

The clause in (1) can be transformed into the following logical equivalent formula.

A₁ AND A₂ AND -------AND Aᵣ

=> B₁ OR B₂ OR------- OR Bₙ

Where => is the implies symbol.

If all the Aᵢ literals (i=1,2,3_ _ _ _ _ n) are true , the formula (2) is true only if at least one of the Bᵢ is true which is the meaning of => (implies).

For formula (1), if all the Aᵢ literals (i=1,2,3_ _ _ _ _ n) are true, their negation are all false so in this case the formula (1) is true only if at least one of the Bᵢ is true.

In Datalog , rules are expressed as a restricted form of clauses called Horn clauses in which a clause can contain almost one positive literal

Hence a Horn clause is either of the form

NOT(A₁) OR NOT (A₂) OR ------- OR NOT (Aᵣ) OR B

The Horn clause in (3) can be transformed into the clause

A₁ AND A₂ AND -------AND Aᵣ => B

Which is written in datalog in the form of a rule?

B:- A₁,A₂, _ _ _ _ _ _ _ _ Aᵣ

In general a query in datalog consists of two `component: a datalog program which is a set of each Xᵢ is a variable or a constant.

A PROLOG or DATALOG system has an internal inference engine. It is used to process and compute the result of queries

Phase II: Interpretation of rules

There are two main methods for interpreting the theoretical meaning of rules.

Proof theoretic, model theoretic

In the first approach of interpretation of rules the fact is considered to be true statement or axioms. Rules are called deductive axioms as they can be used to construct proof that derive new facts from existing facts. In the second method of interpretation we have given an infinite domain of constant values which we assign to a predicate for each combination of values for arguments. In the following example for the predicates SUPERVISE and SUPERIOR .This interpretation assigns truth value for each possible combination for the argument values for the above said predicates. The following example shows how to derive information based on given facts and rules:
**Rules**

1. SUPERIOR(X,Y) :- SUPERVISE(X,Y).
2. SUPERIOR(X,Y):- SUPERVISE(X,Z), SUPERIOR(Z,Y).

**Interpretation**

Known fact:
SUPervise (rakesh, mukesh).
SUPervise (rakesh, sanjay).
SUPervise (rakesh, sunil).
SUPervise (deepak, sumit).
SUPervise (deepak, rishi).
SUPervise (mohit, rakesh).
SUPervise (mohit, deepak).

All the facts defined above are assumed to be true.

Derived Facts:
1. SUPERIOR(rakesh, mukesh).
2. SUPERIOR (rakesh, sanjay).
3. SUPERIOR (rakesh, sunil).
4. SUPERIOR (deepak, sumit).
5. SUPERIOR (deepak, rishi).
6. SUPERIOR (mohit, rakesh).
7. SUPERIOR (mohit, deepak).
8. SUPERIOR (mohit, mukesh).
9. SUPERIOR (mohit, sanjay).
10. SUPERIOR (mohit, sunil).
11. SUPERIOR (mohit, sumit).
12. SUPERIOR (mohit, rishi).

For the derived facts (1-7) the interpretations are as follows: All the above interpretations are true based on the above given facts and rules and the remaining combinations are false. For example if we say SUPERIOR (Deepak, sanjay),it is false as there are no any facts satisfying SUPERIOR (Deepak, Z) and SUPERVISE (Z, deepak).

**References**


[5].Evaluating queries in deducTive databases by generating EL Lozinskii - Proceedings of the 9th international joint conference on …, 1985 - dl.acm.org

[6].Foundations of deducTive databases and logic programming J Minker - 2014


