NOVI SAD J. MATH.
VOL. 30, NO. 2, 2000, 133-140

Proceedings of the TARA 2000 Conference
Novi Sad, Yugoslavia, September 6-7, 2000

PROJECTING DEDUCTIVE DATABASES WITH CWA MANAGEMENT IN BASELOG-SYSTEMS

Biljana Radulović[1], Petar Hotomski[1]

Abstract. In the paper is described a Baselog-system for the deductive databases as an extension of ADT (Automatic Theorem Proving) Resolution System. Flexible binding of the open-world concept in ADT-system and closed-world reasoning in Datalog and Prolog has been realized by means of introducing the CWA-rule and implementing CWA-controller into ADT system. Prolog and Datalog defects related to semantically incomplete bases connected with the negation problem and the CWA-principle have been eliminated. The predicates whose base is semantically complete are treated as in Datalog or Prolog systems, while the predicates whose base is incomplete are being treated as in ADT-system.

AMS Mathematics Subject Classification (1991): 68T35, 68T15

Key words and phrases: Closed-world reasoning, Deductive databases, Datalog, Prolog, Automatic Theorem Proving.

1. Introduction

In [2], [6] and [7] is described Datalog, the Logic Programming Language in the field of the database that is implemented in the post-relation software for the database management system PROGRESS. Datalog works on the CWA-principle, respectively by adopting the Closed World Assumption. The CWA-principle declares ([2], [8]):

Definition (The CWA - principle) *If a fact does not logically follow from a set of Datalog clauses, then we conclude that the negation of this fact is true.*

For knowledge databases is also characteristic the open world assumption. In the open world regime work classic systems for the automatic theorem proving, especially, ADT system [1], [3]. The knowledge bases contain limited knowledge segments from a certain field. They can be incomplete, i.e. they do not present total relevant knowledge. The application of closed world concept on such databases can bring wrong answers to the asked questions. Because of that the pure concept of the closed world can not be applied for the databases used

[1]University of Novi Sad, Technical Faculty "M. Pupin", Djure Djakovića, b.b. Zrenjanin, Yugoslavia, email: { bradulov, hotomski} @ptt.yu

in the education software.

Example. Let there be the following information in the base:

1) The straight lines A and B are parallel.

2) The straight lines B and C are parallel.

Let there be no knowledge of parallelness transitivity of straight lines.

To the pupil's question: "Are the straight lines A and C parallel?" if the education software database is examined as the closed world, the system will generate the negative answer "They are not". That answer is incorrect and it misinforms pupils, what can not be allowed. If the same base is being examined as the open world, the system will answer "It is unknown to me", respectively, "uncertain", what is from the pupil's point of view acceptable. On the other hand, the resolution systems possess theoretical strength for deriving every conclusion whenever it is a logical consequence of the given premises [3] and give correct answers referring to the given starting premise. Universal resolution systems, from a theoretical aspect, support in full the work with databases, but they show a practical deficiency. It can be seen in fact that because of endeavoring to get a semantically expected answer, it is necessary to give a complete space description where the solution is claimed. In the database area it is, for example, exposed through the necessity of proposing the following axiom:

$$t \neq t_1 \wedge t \neq t_2 \wedge \ldots \wedge t \neq t_n \Rightarrow \neg P(t) \tag{1}$$

where $t_1, \ldots, t_n$ are the relation database tuples, and $P(t)$ means that the tuple t is belonging to the database relation. As can be seen, already for a small number of tuples in the database, this axiom is long, so this theoretical possibility is omitted in practical applications. Both in Datalog and Prolog attempts are made to overcome this shortcoming in specific ways. In Prolog it is the strategy of definite failure [3], [1] and in Datalog the CWA-principle [8]. However, none of these solutions can satisfy education needs in full, for the following reasons. In reference to the possible user's questions, there are the following options:

a) the answer to the question is deducible from the base,

b) the answer to the question is not deducible from the base, where there are two possibilities:

b1) the answer needs to be affirmative,

b2) the answer needs to be negative.

In a), when the answer is deducible from the base, it can be found and presented to the user either in Prolog, Datalog or ADT-system. Specificities are being reflected in b). According to the adopted CWA-assumption in Datalog, respectively the definite failure concept in Prolog, there are possible incorrect or indefinite answers. So in b1) Datalog can generate the incorrect answer NO, while Prolog's answer "NO" can be interpreted as "uncertain". In b2) Datalog answer "NO" is correct, and Prolog's answer "NO" can be interpreted as "NO". In both cases b1) and b2) ADT gives answer "uncertain".

We observe that in educative sense Datalog according to b1) is not satisfactory, while Prolog and ADT give acceptible, but uncertain answers. In b2) Datalog gives correct and precise answer, while Prolog and ADT give inadequately precise answers.

From the educative aspect it is desirable to lessen the indefiniteness of the system answer and it is necessary to eliminate the unallowed answers. Otherwise, there is a need to keep the definiteness present in Datalog for b2) and eliminate the unallowed answer from b1). By implementing Baselog-system projected on the list of CWA-predicate and the CWA-rule, a flexible concept is realized. By this system all predicates which are in the CWA-list are treated as Datalog in closed-world, while all the other predicates are treated in open world, i.e. it works as ADT. At that, it is free of Prolog defects in reference to the negation treatment and the definite failure concept.

2. The CWA-rule and CWA-controller in Baselog-system

The basis for Baselog-system make the following components: The CWA-predicate list, which is a part of the program; the CWA-rule and CWA-controller by which is enlarged ADT resolution system. Every literal of the form $R(w_1, \ldots, w_m)$ where R is predicate name mentioned in the CWA-predicate list, and $w_1, \ldots, w_m$ are arguments, the Baselog-system will treat in the closed system regime, while all the other predicates that are not in the CWA-predicate list, the system will treat in the open world regime. Here, the CWA-controller of Baselog-system uses the CWA-rule, formulated in the following way.

2.1. The CWA - RULE

Let D be a clause of the form $L_1 \vee L_2 \vee \ldots \vee L_p$ and let $L_i, 1 \leq i \leq p$ be a literal of the form $R(w_1, \ldots, w_m)$, where the predicate R is declared as a CWA-predicate. If $R(w_1, \ldots, w_m)$ can be unified with no one base element, then $R(w_1, \ldots, w_m)$ will be deleted from the clause D. If there exists a unificator for L_i and some element from the base, then the clause D is not changed, and there is no deleting.

The proof of the CWA-rule correctness: On the fullness of the resolution rules, it is enough to prove that whenever the CWA-rule can be applied, it generates a resolvent that would generate in a pure resolution procedure with the aid of the adding axiom for the world closeness in relation to the actual predicate.

Let on the level l be generated by resolution the resolvent:

$$D : L_1 \vee L_2 \vee \ldots \vee L_p \tag{2}$$

where the literal $L_i, 1 \leq i \leq n$ is of the form $R(w_1, \ldots, w_m)$, where R is declared as the CWA-predicate. In the system that uses only resolution, the adding axiom is necessary to provide the treatment of the predicate R in the closed world. There are two possible cases:

a) the R predicate does not appear in the base,
b) the R predicate appears in the tuples $t_1, t_2, \ldots, t_k, k \leq n$ (n is the number of all tuples in the base).

In the case a) the adding axiom is:

$$(Ax.1) \qquad \neg R(t)$$

for every t of the form $(x_1, \ldots, x_m)$, where $x_i, i = 1, \ldots, m$ are variables. In the case b) the adding axiom is:

$$(Ax.1') \qquad t \neq t_1 \wedge t \neq t_2 \wedge \ldots \wedge t \neq t_k \Rightarrow \neg R(t)$$

in the form of the clauses:

$$(Ax.1") \qquad t = t_1 \vee t = t_2 \vee \ldots \vee t = t_k \vee \neg R(t)$$

The case a)

By resolution from D and Ax.1 is derived the resolvent:

$$(D \setminus \{L_i\})_\theta : L_{1\theta} \vee \ldots \vee L_{i-1\theta} \vee L_{i+1\theta} \vee \ldots \vee L_{p\theta}$$

where $\theta = \{w_1/x_1, \ldots, w_m/x_m\}$ is a unificator for t and $(w_1, \ldots, w_m)$. As Ax.1 and D do not contain common variables (it can be always achieved by redesignating), it follows that the substitution θ does not change the literals $L_1, \ldots, L_p$. Because of that the resolvent is

$$(D \setminus \{L_i\})_\theta : L_1 \vee \ldots \vee L_{i-1} \vee L_{i+1} \vee \ldots \vee L_p$$

Just that is obtained by applying the CWA-rules on D. Really, in this case for none base element there is a unificator for $R(w_1, \ldots, w_m)$ (because the R predicate in the base). The proof of the case a) is finished. □ The case b) By resolution from D and Ax.1" one derives the resolvent:

$$L_{1\theta} \vee \ldots \vee L_{i-1\theta} \vee L_{i+1\theta} \vee \ldots \vee L_{p\theta} \vee t_\theta = t_{1\theta} \vee \ldots \vee t_\theta = t_{k\theta}$$

where θ is the unificator for t and $(w_1, \ldots, w_m)$ where that replacement effects only onto the rest Ax.1" and does not effect the tuples $t_1, \ldots, t_k$ (they contain constants), so the resolvent is:

$$L_1 \vee \ldots \vee L_{i-1} \vee L_{i+1} \vee \ldots \vee L_p \vee t_\theta = t_1 \vee \ldots \vee t_\theta = t_k$$

Now there are two possible cases: b1) and b2).

The case b1)

None of the equalities is satisfied, i.e. $t_\theta \neq t_1, \ldots, t_\theta \neq t_k$, so the derived resolvent is:

$$L_1 \vee \ldots \vee L_{i-1} \vee L_{i+1} \vee \ldots \vee L_p$$

Indeed, this is obtained by applying the CWA-rules onto D, because there is no unificator, only for the tuples $t_1, \ldots, t_k$ from the base.

The case b2) Any of equalities is satisfied, i.e. $t_\theta = t_j, (j \in \{1, 2, ..., n\})$ is worth. Then in the base there exists the element $R(w_1, \ldots, w_m) \equiv R(t_\theta) \equiv R(t_j)$, so deleting from the CWA-rules does not apply in this case.

With this, the proof is complete. □

2.2. The principle of CWA-rule embedding into resolution procedure

The CWA-rule will be applied on the given clause (resolvent) only if on that clause one can not apply the resolution rule, and all the conditions for using the CWA-rules are fulfilled.

This principle depends neither on the concrete resolution form, nor on the concrete retrieval strategy, so it can be applied onto every of them. General block scheme of the resolution is modified without disturbing basic structure by adding the CWA-controller module.

The advantage is given to the resolution because in the program that uses database, whose elements are considered as axioms, can also exist special, so-called auxiliary, axioms. By auxiliary axioms it is possible to introduce also the predicates that do not exist in the database, and which are connected with database predicates.

For example, let the database elements contain the predicate PAID: PAID(*ann*), PAID(*bertrand*), PAID(*charles*). By auxiliary axiom one can introduce the predicate DEBTOR:

$$\text{DEBTOR(X)} \Leftrightarrow \neg\text{PAID(X)}$$

So, the introduced predicates may be declared or they may not be declared as the CWA-predicates. In fact, if the base predicate is declared as the CWA, that is indirectly reflected onto the other predicates that are connected with it. □

2.3. The CWA-controller as a module for widening ADT system

The CWA-rule has been used for a module that widen the existing ADT system, that is founded onto the OL resolution with marked literals [3], [1]. The CWA-controller for this kind of resolution works in the following way:

If the last literal of the central clause, whose form is $R(t_1, \ldots, t_n)$, can not be refuted by the auxiliary axioms, and can not be unified with any base element, then the CWA-rule was applied and that literal will be deleted.

Then the shortening and compressing operations are applied onto the clause thus identified, these operations being otherwise applied in the process of OL resolvent generation. So, the generated clause presents a new resolvent that is taken as a new central clause for generating resolvents on the next level. The CWA-controller as an intrinsic module activates only for those predicates that are declared in the CWA-predicate list. For all other predicates the system works as an ADT system without the CWA-controllers. The basic concepts of Baselog-system are: The BAX-type axioms, by which are presented the database elements (tuples) in the system (analogy with the EDB-predicates in [2], [8]);

the auxiliary axioms are the AX type axioms (analogy with the IDB-predicates in [2], [8]); the CWA-predicate list; the queries.

The BAX type axioms are taken for presenting database elements. Every database element gets an axiom status. The connection for the internal result of work system with semantical meaning:

If an inquiry predicate is declared as the CWA-predicate, then internal answers of Baselog-system have the following meaning:

1. The answer of the system **success = 1**, means the affirmative answer to the inquiry, respectively "YES".
2. The answer of the system **success = 0** and has no new resolvents, means the negative answer to the inquiry, respectively "NO".
3. The answer of the system **success = 0** and it is possible to generate more resolvents, but the process interrupts because of space-time limitations, that means that the system answer is "*uncertain*".

If an inquiry predicate is not declared as the CWA-predicate, then Baselog-system answers are interpreted as in the ADT system in the following way:

1. The answer of the system **success = 1**, means the affirmative answer to the inquiry, respectively "YES'.
2. The answer of the system **success = 0** and has no new resolvents, means "*undeducible*" (can not be proved), but does not mean NO.
3. The answer of the system **success = 0** and it is possible yet to generate a resolvent, but the process is interrupted because of space/time limitations, it means "*uncertain*".

The database designer in Baselog-system, choosing the CWA-predicate list, manages the system behavior.

The determination of CWA-predicates is performed as follows:

If a relation is completely described in the database, respectively, the database is semantically complete in reference to the predicate that corresponds to that relation, then such predicate declares as a CWA-predicate.

Concisely, the choice principle of the CWA-predicate is:

A predicate declares as the CWA-predicate iff the database is complete in reference to the relation described by the predicate.

3. Examples for the comparison of Baselog with Datalog and Prolog

In [4] and [5] are mentioned the examples of Baselog-system work on a real knowledge base with one predicate. In practice, bases contain many different predicates. Declaring which predicate will be the CWA-predicate depends on the base content, respectively on its completeness in reference to that predicate (the attribute it describes). It is illustrated by the following more complex example.

Example. There are the following elements in the database:

MEDITERRANEAN_STATE(Portugal),.,MEDITERRANEAN_STATE(Morocco), respectively are cited the names of all **mediterranean** states.
HAS_EXIT_TO_SEA(England),.,HAS_EXIT_TO_SEA(India), respectively, are cited names of **some** states that have exit to sea.
CAPITAL_MED_STATE(Athens,Greece),.,CAPITAL_MED_STATE(Rabat, Morocco) respectively,
are cited the names of the capitals of **all** Mediterranean states.
IS_CAPITAL(London,England),.,IS_CAPITAL(Bonn,Germany), respectively, are cited the names of the capitals **only for some** world states. Analyzing the database contents the designer concludes that the attributes are: **is_a_mediterranean_state and the_capital_of_the_given_mediterranean_state** fully described, respectively, the database is complete in reference to them. Therefore, the adequate attributes: MEDITERRANEAN_STATE and CAPITAL_MED_STATE declare for the CWA-predicates.

The other predicates: HAS_EXIT_TO_SEA and IS_CAPITAL must not be declared for the CWA-predicates, because the database in reference to them is not complete. On the basis of the database thus designed and the CWA-predicate list, Baselog-system will continually give correct answers.

CLOSED WORLD

To the question:

Is Kuwait a Mediterranean state?

the answer of Baselog, Prolog and Datalog is NO. Therefore, in the regime of closed world the answers of all three systems agree and they are correct in consideration of the fact that the base is complete.

OPEN WORLD

Disagreement with Datalog

To the question:

Does Finland has an exit to sea?

respectively, in the Baselog form:~ HAS_EXIT_TO_SEA(Finland)&
the answer of Baselog-system is UNCERTAIN (UNDEDUCIBLE) because that fact is not in the database, Datalog answer is NO, and Prolog answer is "NO".

In this case the Datalog answer is incorrect and educationally unacceptable.

Disagreement with Prolog

Let the predicate **a_continental_state** is defined (determined) by the next axiom:

CONTINENT_STATE(X1) :- not HAS_EXIT_TO_SEA(X1).

To the question: "Is Finland a continental state?" Prolog system gives the answer YES, which is wrong from the semantic aspect. This answer Prolog gives as in the generated goal **not HAS_EXIT_TO_SEA(Finland)**, the element **HAS_EXIT_TO_SEA(Finland)** suffers definite failure (that fact is not in the database), so it is accepted that the goal is satisfied. Because of the CWA-principle, Datalog will give the same answer as Prolog, i.e., YES which is, also wrong. The Baselog answer is UNCERTAIN (UNDEDUCIBLE), with

consideration that "CONTINENT_STATE" and "HAS_EXIT_TO_SEA" are not declared for the CWA-predicates, which is educationally acceptible.

4. Conclusion

Baselog-system concept and its program implementation enabled the integration of good properties of Datalog, Prolog and ADT system, and so is realized a more flexible system in reference to the work in the closed, respectively open world. Specific needs in the development of the education computing software for the work with databases demand just the development and application of such a system, with performances superior compared with Datalog, Prolog and ADT-system considered separately. The directions of further development in Baselog-system are related to the development of a special user's interface for giving semantic answers and enabling one to use Baselog-system with no need of knowing its internal mechanisms. Also, it is possible to realize extracting only the wanted data from the database, and further to work out the technique to form BAX type axioms from the existing databases.

References

[1] Berkovic I., The Deductive Bases for Development of the Descriptive Languages for the Logical Programming, Ph.D. Thesis, University of Novi Sad, 1997 (in Serbian).

[2] Ceri S., Gottlob G., Tanza L., What You Always Wanted to Know About Datalog (And Never Dared to Ask), IEEE Transactions on Knowledge and Data Engineering, Vol. 1, No. 1 March 1989, 146-167.

[3] Hotomski P., Systems of Artifical Intelligence, Technical Faculty "Mihajlo Pupin", Zrenjanin, 1995 (in Serbian).

[4] Radulovic B., Database Projecting in the Field of Education Computer Software, Ph.D. Thesis, University of Novi Sad, 1998 (in Serbian).

[5] Radulovic B., Hotomski P., Database projecting in the Baselog-system, VII Conf. "Informatics in education and new information technologies", Novi Sad, 1997, 71-77. (in Serbian)

[6] Stonebraker M., Rowe L.A., Hirohama M., The Implementation of POSTGRES, IEEE Transactions on Knowledge and Data Engineering, Vol. 2, No. 1, March 1990, 125-142.

[7] Ullman J., Assigning an Appropriate Meaning to Database Logic With Negation, www- db.stanford.edu/pub/papers/negation.ps, 1994.

[8] Ullman J., Principles of Databases and Knowledge-Base Systems, Vol. I, Computer Science Press, New York, 1988.