



Journal of Applied Sciences

ISSN 1812-5654

science
alert

ANSI*net*
an open access publisher
<http://ansinet.com>

A New Approach for Termination Analysis of Active Rules

H. Belbachir and N.S. Ougouti

Department of Data Processing, U.S.T.O-LP 1505 El M' Naouer, Oran, Algeria

Abstract: This study presents a new static approach for termination analysis of the active rules. It consists of the detection of cycles in a graph says dependences graph. This last is built by taking into account the triggering of rules by other rules, influence of rule's actions on triggered rule's conditions and satisfiability of rule's conditions. A comparative study of the method suggested with the most known approaches was made thus showing our contribution in the field considered.

Key words: Static termination method, dependences graph

INTRODUCTION

Databases Management Systems (DBMS) are currently used in various fields such as the CAD, office automation, software genius, geography, medicine, the speech processing etc., they constitute essential elements of the information systems.

Several generations of DBMS were born, while starting with the hierarchical and networks DBMS, followed by the relational DBMS and then the Oriented Object DBMS.

These DBMS remain nevertheless passive because they carry out operations only in response to explicit requests coming from the application programs or the users.

An approach appeared consisting in making these systems active, i.e., able to react to significant situations (events) by executing the suitable actions in an automatic way, without the intervention of the user or the application. This capacity of reaction is modelled in general by active rules of type ECA (Event, Condition, Action) whose semantics is: When the event E occurs if the condition C is checked then execute action A.

The applications of the active databases are very varied, they can manage the problems of integrity constraints, they can deal with the management of the sights, the versions or the evolution of the diagrams.

The introduction of the active rules into DBMS produced new problems. Among these problems that of no termination of execution of rules. Indeed, it is ensured that the execution of a set of rules finishes, if for any state of the database and for any initial starting operation, the rules cannot be activated indefinitely. Thus, it is necessary to take measures to prevent against an infinite execution of the system. Several researches was started to try to give a solution to this problem.

We present in this study, our contribution to the resolution of the problem of no termination of rule systems by proposing a method of static analysis which is based on the realisation of a particular graph called Dependences Graph (DG) and which deduces the no termination of rules when cycles appears in this graph.

State of the art of termination analysis of active rules:

There are two kinds of termination analysis methods :

- Static analysis methods which operate during the compilation, such as the programmer can know in advance possible problems in the behaviour of rules^[1-9]
- Dynamic analysis methods^[10-13], which operate during the execution, to detect possible loops which can cause no termination of the system and to supervise the behaviour of certain suspect rules detected during the static analysis.

In what follows, we briefly present the most important static analysis methods which exist in literature:

One of the first works in this field is that of Aiken *et al.*^[1] who are the first to introduce the concept of triggering graph. This last is a graph where the nodes represent the rules and the arcs represent the triggering relation between these rules. If this graph is acyclic, the termination of the system of rules studied is guaranteed, but one can nothing say in the contrary case.

This idea was improved thereafter by Karadimce and Urban^[9] which propose to use in addition to the triggering graph, a triggering formula. This last is associated to an arc (Ri, Rj) and represents the conjunction of the conditions of Ri and Rj. If this formula is unsatisfiable, the conditions of Ri and Rj cannot be satisfiable at the same

time and thus the execution of rule R_i cannot Trigger the rule R_j and the arc (R_i, R_j) does not exist. If the graph obtained is acyclic then the termination is guaranteed.

Baralis and Widom^[3] based their study on a activation graph which is similar to the triggering graph with a small difference since in this graph the nodes represent the rules and the arcs (R_i, R_j) indicate that a rule R_i can activate the condition of another rule R_j (the action of R_i returns satisfiable the condition of R_j). The interaction between the action of a rule and the condition of another rule is detected thanks to an algorithm called propagation algorithm which uses an extension of the relational algebra, to propagate the rule's action through the condition of another rule. As the two preceding methods, the analysis of the termination is done while being based on the search for cycles on the activation graph, if this graph is acyclic, the termination is guaranteed.

One year later Baralis *et al.*^[4] presented an approach based on the analysis of the two graphs i.e. the triggering graph and the activation graph. They use an algorithm of rules reduction. If at the end, the set of rules is empty then the termination is guaranteed.

The approach of Lee and Ling^[5] represents an extension of the work of Karadimce and Urban, because they further push the analysis by making an additional study on the cycles met to try to see if cycles will stop at the end of a certain number of executions.

Zimmer *et al.*^[6] presented an analysis of rule termination based on the ordinary petri nets. They showed how the execution of a system of rules is modelled by the petri nets and how, by exploiting the reachability graph and the coverability graph, one can have information on the termination of these rules.

Kokinaki^[7] used the parameterized petri nets for modelling and analysing the active rules, the presence of cycles in this graph indicates a system no termination.

Lastly, Rabih^[8], Karadimce and Urban^[9] thought of reducing rules ECA to a terms rewriting system and then applied techniques of termination already established for these systems to prove termination of active rules.

Presentation of our approach for termination static analysis: The approach suggested is a static analysis method based on a graph model. It takes into account the influence of the actions on conditions and on the other hand the satisfiability of rule conditions.

The built graph is called Dependences Graph (DG) where the nodes are rules and there is an arc between two rules R_i and R_j , if on the one hand rule R_i contains in its action part, an event triggering R_j and on the other hand if the action of R_i or a rule precedent R_i in the graph does

not make false the condition of R_j (otherwise the R_j rule cannot be executed because its condition will not be checked) and if the condition of R_i or a rule precedent R_i does not contradict that of R_j (otherwise the R_j rule cannot be executed because its condition cannot be satisfied at the same time with condition of R_i or of a rule precedent R_i). The termination is guaranteed if there is no cycle in the graph considered.

The construction of the dependences graph lean on three concepts:

- Triggering concept.
- activation concept.
- Triggering formula concept.

Construction of the dependences graph: The dependences graph is a directed graph, composed of nodes representing the system rules and arcs between couples of rules. To trace an arc between two rules R_i and R_j , it is necessary to follow an algorithm which tests in the order the three already quoted concepts.

1. The first test consists in checking if R_i triggers R_j , i.e. if there exists in the action part of R_i an event which trigger the R_j rule.
2. If R_i triggers R_j , the second test i.e., the test of the activation concept, consist in checking, if the action of R_i activates the condition of R_j (i.e., returns it satisfiable), three possibilities are presented:
 - The action of R_i activates the condition of R_j , thus the arc between R_i and R_j is added to the dependences graph.
 - The action of R_i forge the condition of R_j i.e., returns the condition of R_j insatisfiable (the R_j rule cannot be executed if R_i were executed), thus the arc between R_i and R_j will not be added to the dependences graph.
 - The action of R_i does not have any influence on the condition of R_j and thus it is necessary to make a back return while following the triggering order to try to find an action which checks the case a) or the case b) with the condition of R_j . In the contrary case, one passes to the test of the third concept.
3. The third test i.e., the test of the triggering formula are made only if the test of the activation concept did not conclude anything, it is acted in order to check the satisfiability or not of triggering formula which consists on the conjunction of the two conditions of R_i and R_j , two cases arise:
 - The triggering formula of R_i and R_j is not satisfiable i.e., that the conditions of R_i and R_j are contradictory and thus the R_j rule cannot be executed if R_i regulates it were executed. In this case the arc

- between R_i and R_j is not added to the graph.
- The triggering formula of R_i and R_j is satisfiable, in this case one makes a back return on the rules triggered before R_i to see whether there is a rule whose condition is contradictory with that of R_j , if it is the case, we do not trace the arc, otherwise we trace the arc between R_i and R_j .

Comparison between our approach and other approaches:

Here we are going to give examples of cases unsolved by certain approaches and which found solution with the method that we propose.

Approach of aiken and hellerstein: By the use of an example, we are going to show that this approach shown a no termination, whereas it is not the case and this is what it is shown by our method.

This example is formed by the following system of rules:

R1: $e1(x, y)$ if $(x > 5)$ then $e2(x, y)$
 R2: $e2(x, y)$ if $(y = 1)$ then $e3(x, y)$
 R3: $e3(x, y)$ if $(x < 5)$ then $e1(x, y)$

The triggering graph obtained by the approach of Aiken *et al.*^[1] is given in Fig. 1.

According to the method of Aiken and Hellerstein, the existence of cycle leads to no termination of the rules system. By considering the rules, one sees that the cycle (R1, R2, R3) is a false cycle because the conditions of R1 and R3 are contradictory.

The algorithm of the approach suggested by us can detect this case (ai and ci, respectively represent the action and the condition of R_i):

R1 triggers R2

- Activation Test: R1.a1 does not contain an update which returns satisfiable or not the R2.c2 condition. There is no back return because there is not rule before R1 in the triggering graph.

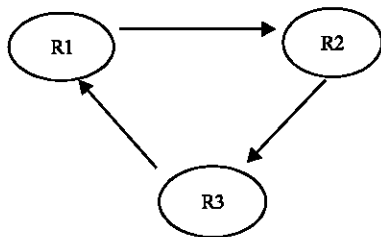


Fig. 1: Triggering graph of example 4.1

- Triggering formula Test: The formula $(R1.c1) \wedge (R2.c2)$ is $((x > 5) \wedge (y = 1))$ and is satisfiable. There is no rule preceding R1, then we trace the arc (R1, R2).

R2 triggers R3

- Activation test: R2.a2 does not contain an update which activates or not the R3.c3 condition. There is no R_k rule in a sequence of execution..... R_k, R2, R3 such as the $R_k.a_k$ action activates the R3.c3 condition.
- Triggering formula test: The formula $(R2.c2) \wedge (R3.c3)$ is $((y = 1) \wedge (x < 5))$ and is satisfiable. There is a R_k rule = R1 in the sequence of execution R1, R2, R3 such as R1 is the first rule met in back return and such as $(R1.c1) \wedge (R3.c3)$ $(X > 5$ and $x < 5)$ is not satisfiable, then we do not trace the arc (R2, R3).

R3 triggers R1

- Activation Test : R3.a3 does not contain an update which activates or not the condition R1.c1, there is no R_k rule in a sequence of execution..... R_k, R3, R1 such as the $R_k.a_k$ action activates or deactivate the R1.c1 condition.
- Triggering formula test: the formula $(R3.c3) \wedge (R1.c1)$ is $((x < 5) \wedge (x > 5))$ and is not satisfiable then we do not trace the arc (R3, R1).

The dependences graph obtained, represented in Fig. 2, does not contain cycles, thus termination of this rules system is guaranteed (Fig. 2). The cycle detected in the approach of Aiken and Hellerstein is a false cycle.

Approach of Baralis, Ceri and Paraboschi: This approach also detects cases of no termination whereas our method proves that it is a termination.

This example is based on the following rules:

R1: $e1(z)$ if $(u = 0)$ then $v := 0$ $w := 0$ $e2(x)$
 R2: $e2(x)$ if $(v = 0)$ then $v := 1$ $w := 1$ $e3(y)$
 R3: $e3(y)$ if $(w = 0)$ then $u := 0$ $w := 1$ $e1(z)$

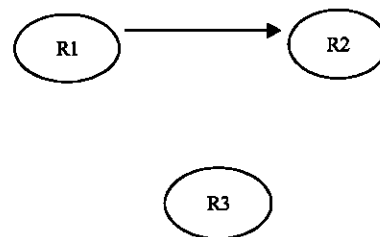


Fig. 2: Dependences graph of example 4.1

The algorithm of analysis proposed by this approach uses the triggering graph TG and the activation graph AG, and is as follows:

Repeat until there is no are more reductions to do If (no arc(R' , R) $\in$ TG) or (no arc(R' , R) $\in$ AG) then Remove R of TG and AG.

By applying this algorithm, the set IR of rules found at the end is nonempty, thus the authors conclude that the execution of this system of rules can not finish, but if it is attentively examined, one can easily detect that the R3 rule cannot be executed because its condition is always evaluated to false because of the action of R2.

Let us apply our approach to the same example:

- R1 triggers R2 and R1.a1 ($v:=0$) returns to true the condition R2.c2 ($v=0$) then we trace the arc (R1, R2)
- R2 triggers s R3 and R2.a2 ($w:=1$) returns to false the condition R3.c3 ($w=0$) then we do not trace the arc (R2, R3)
- R3 triggers R1 and R3.a3 ($u:=0$) returns to true the condition R1.c1 ($u=0$) then we trace the arc (R3, R1)

The dependences graph obtained does not contain cycles, then the termination of this system of rules is guaranteed (Fig. 3)

Approach of Baralis - widom: We will prove with an example that the approach of Baralis-widom shown a termination of rules, whereas actually this latter is not guaranteed.

This example is based on the following rules:

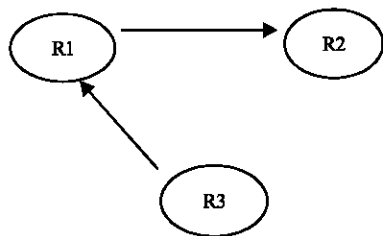


Fig. 3: Dependences graph resulting from example 4.2

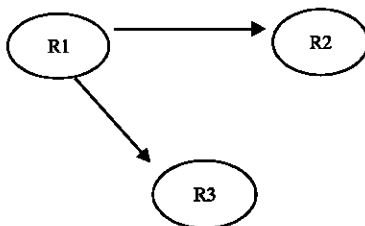


Fig. 4: Activation graph of example 4.3

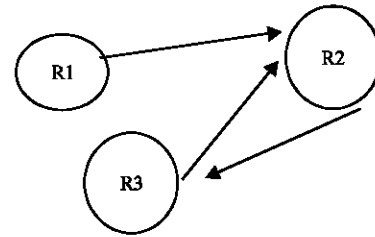


Fig. 5: Dependences graph resulting from example 4.3

R1: e1 if ($x=0$) then $y=0$; e2.

R2: e2 if ($y \geq 0$) then e3.

R3: e3 if ($y < 5$) then e2.

The activation graph obtained does not contain cycles, therefore the termination is guaranteed according to authors (Fig. 4). But if one looks at well, the two rules R2 and R3 triggers mutually and thus the termination is not guaranteed.

By applying the method suggested by us, one obtains the dependences graph which contains a cycle not detected by the approach of Baralis-Widom and thus the termination of this system of rules is not guaranteed (Fig. 5)

Advantages of the approach suggested: We saw that the majority of the static approaches used a graph model on which the termination analysis is done, some used the triggering graph, others used the activation graph, or a refined triggering graph, where the concept of triggering formula is introduced. However all these methods were considered to be too conservatives and examples proving their weakness were given for many of them. But, one can say that these methods nevertheless had the merit to explain the problem of no termination well and to introduce concepts which served as other studies. But the fault of these methods is that each one was based on a unique concept without checking the others (triggering, activation, triggering formula).

The most basic idea of our approach consists in using and exploiting the force of these three concepts at the same time. Moreover, our method does not stop in the test and the checking with a couple of rules but made a back return in the rules started previously to try to find a rule which by the means of one of the concepts does not allow the execution of the currently triggered rule. In this way the graph obtained cannot contain false cycles.

The triggering concept will be used to limit the number of tests between couples of rules since it will be done only between the triggering-triggered rules and to

find the rules of the execution sequence in the case of a back return.

The activation concept permits to see whether the condition of the triggered rule is not influenced by the update of the action of the triggering rule or one of the rules preceding it.

The triggering formula concept permit to know if the condition of the triggered rule is contradictory with that of the triggering rule or one of the rules preceding it in the execution sequence, which would block its execution.

The use of these three concepts and the various back returns represent the strong points of our approach and returns the graph generated for the study of the termination very powerful semantically.

CONCLUSIONS

This study presents our contribution in the field of static termination analysis of active rules, since it initially summarizes principal work which was made in this field on static approach and proposes a new algorithm for the resolution of this problem. This new approach exploits the syntactic aspect of rules and draws its force from the semantic relations which exist between the actions and the conditions on the one hand and between the conditions between themselves on the another hand. It detects the rules which can be the cause of a problem of no termination and can thus be a part of an interactive tool for the rule specification of an active database.

We tried to compare our approach with other existing approaches and proved on some examples that it can cure the limits of these last.

REFERENCES

1. Aiken, A., J. Widom and J.M. Hellerstein, 1992. Behaviour of database production rule termination, confluence and observable determinism. In: ACM SIGMOD Data, San Diego, California, pp: 59-68.
2. Karadimce, S.P. and S.D. Urban, 1996. Refined triggering graphs: A logic-based approach to termination analysis in an active object-oriented database. In: Proceeding of International Conference on Data Engineering, (ICDE), New-Orleans, Louisiana, pp: 384-391.
3. Baralis, E. and J. Widom, 1994. An algebraic approach to rule analysis in expert database systems. In: Proceeding of 20th Int. Conference Very Large Data Bases, Santiago, Chile, pp: 475-486.
4. Baralis, E., S. Ceri and S. Paraboschi, 1995. Improved Rule Analysis by means of triggering and activation graphs. In: Proceeding of International Workshop Rules in Database Syst., (RIDS), Athens, Greece, pp:165-181.
5. Lee, S.Y. and T.W. Ling, 1999. Unrolling cycle to decide trigger termination. Proceeding of 25th VLDB Conference, Edinbourg, Scotland, pp: 483-493
6. Zimmer, D., R. Unland and A. Meckenstock, 1996. Rule termination analysis based on petri nets. Internal Res. Rep., Paderborn Univ., pp: 1-17
7. Kokkinaki, A. I., 1998. On using multiple abstraction models to analyse active databases behaviour. Res. Rep. Internal, Chipre Univ., pp: 1-8
8. Rabih, Y., 1998. Démarche générique de modélisation et de simulation pour la vérification du comportement de règles actives. Ph.D. Thesis, Doctoral School Sciences for the Engineer of Clermont-Ferrand, Blaise Pascal Clermont II Univ., pp: 47-84
9. Karadimce, S.P. and S.D. Urban, 1994. Conditional term rewriting as a formal basis for analysis of active database rules. Proceeding of 4th International Workshop. Issues in Data Eng.: Active Database Systems, RIDE-ADS 94, Houston, pp: 156-162
10. Behrends, H., 1994. Simulation-based debugging of active databases. Tech. Rep., Oldenburg Univ., pp: 1-9
11. Diaz, O., A. Jaime and N. Paton, 1993. DEAR: A Debugger for active rules in an object-oriented context. In: Rules in Database Syst., pp: 180-193
12. Fors, T., 1995. Visualization of rule behaviour in active databases. Internal Res. Rep., Univ. Skövde, Sweden, pp: 1-20.
13. Kappel, G., G. Kramler and W. Retschitzegger, 2001. TriGS Debugger a tool for debugging active database behavior. Res. Rep. Internal, Linz Univ., Austria, pp: 1-12.