Constraint Databases Technology for Polynomial Models Diagnosis

M. T. Gómez and **R. Ceballos** and **R. M. Gasca** and **C. Del Valle** [1]

Abstract. Model-based Diagnosis allows the identification of the parts which fail in a system. The models are based on the knowledge of the system to diagnose, and they can be represented by constraints associated to the components. The variables including in these constraints can be observable or non-observable, depending on the situation of sensors. In order to obtain the minimal diagnosis in a system, an important issue is related to find out the minimal possible conflicts in an efficient way. We consider that Constraint Databases represent an excellent approach in order to solve this problem in complex systems, where a tuple in a relational database could be replaced by a conjunction of constraints.

In this work we have used a novel logical architecture of Constraint Databases which has allowed us to obtain these possible minimal conflicts by means of a standard query language though the information is stored in a conventional relational database. Moreover, we have considered Gröbner bases as a projection operator to obtain the minimal possible conflicts of a system.

1 Introduction

In industrial production, the faults produced in components and processes can cause undesirable stops and damage in the systems with the consequent cost rise and production decrease. It is also necessary to take into account the negative impact in the environment that these faults can produce, has to be avoided. For this reason, in order to keep the systems within the desired security, production and reliability levels, we use mechanisms which allow us the detection and diagnosis of the faults produced in the systems.

Diagnosis allows us the identification of failures in a system, and with Constraint Databases (CDB) technology [14, 17, 8] we are able to make persistent the information and models. Our proposal is based on DX [2] approaches and in other works as [16, 3]. These works were proposed to find out the discrepancies between the observed and correct behaviors of the system. There is another paper related to our work [18] that introduces an algorithm for computing minimal diagnosis for tree structured systems. Our proposal presents how to diagnose any polynomial system without backward propagation because this propagation is not possible in all cases.

The models centered on mechanisms describe the systems by means of input-output relations. Model-based diagnosis is considered by a pair {SD,OBS} where SD is the system description and OBS is a set of values of the observable variables. In complex systems it manages data from various observations, equations of the system description, and contexts at an enormous rate. In engineering applications the storage of these data and query processing are often overlooked.

The original idea of this paper is to combine the power of CDBs [13, 9] with the data treatment of diagnosis. We are able to improve the efficiency in some phases of the model-based diagnosis with CDBs. In order to improve the detection of possible minimal context conflicts, we use a program implemented in Java^{TM} and it uses SQL (Standard Query Language). SQL makes easier the treatment and extraction of a great quantity of information.

We concretely tackle the determination of the possible minimal conflicts of a system. A conflict is a set of assumptions where, at least, one of them must be false. The assumptions are about behavioral modes of components. GDE coupled with an ATMS [4] as inference engine uses previously discovered conflicts to constrain the search in the candidate space. Conflicts are identified in the process of constraint propagation through recording dependencies of predicted values given the system description and the observations. A conflict is minimal if none of its subsets is a conflict. The most important disadvantage of using this approach is the large number of possible conflicts ($2^n - 1$) that must be study, n being the number of components.

In last years, this problem has been an active area of research, in order to find the minimal conflicts. There are different ways to find out all the possible minimal conflicts. One of these ways is preprocessing the initial model, independence and incrementally [5]. Others use the calculation of the minimal chains which can be evaluated [15], or by means of symbolic processing techniques (Gröbner bases) of the initial model [7].

Our work presents a novel methodology that uses CDBs technology for the determination of possible minimal conflicts. A tuple in a relational database could be replaced by a conjunction of constraints from a standard query language. The objective is promoting the advantages of CDBs technology and its usage in a real problem of industrial diagnosis.

In our paper, the model is stored in a Polynomial CDB, using standard query language to store and obtain the necessary information. A preprocessing step reduces, in a significant way, the number of possible contexts to be treated using symbolic techniques. This technique for elimination is Gröbner bases that correspond to our projection operator. It eliminates the non-observable variables from the constraints of the different contexts in a previous step.

The relational model is not able to represent the scientific and engineering data. However, CDBs are usually applied to spatial-temporal data [12, 10] and only a few are applied to engineering data. But in this work we use CDBs as an easier way to represent, in a compact form, components behaviors describe by constraints.

Our paper has been organized as follows: Section 2 reviews defi-

[1] Departamento de Lenguajes y Sistemas Informáticos, University of Seville, Computer Engineering Superior Technical School, Avenida Reina Mercedes s/n 41012 Sevilla (Spain) email: {mayte,ceballos,gasca,carmelo}@lsi.us.es

nitions and notations to allowing to formalize the subsequent operations. In Section 3 we show an example to prove our solution and the use of CDB architecture to treat the information. Section 4 exposes how we solve the example with CDB that improve other solutions. Finally we present our conclusions and the future works in this research line.

2 Definitions and notation

The definitions and notation used are based on the concepts proposed in the diagnosis community (DX). The objective is that the synergy of both approaches will produce better diagnosis results. To introduce our work it is necessary the use of the following definitions and notation:

Definition 1. The System Polynomial Model (SPM): It can be defined as a finite set of polynomial equality constraints P which determine the system behavior. This is done by means of the relation between the system non-observable variables (Vnob) and the observable variables (Vob) which are directly obtained from sensors that are supposed to work correctly. A SPM is the tuple (P,Vob,Vnob).

Definition 2. Diagnosis Problem (DP): It can be defined by means of a tuple formed by a SPM and an Observational Model. The result of this problem will be a set of elements that belong to the set of the system faults which reflect, in a minimal way, the information of the possible failing components DP(SPM,OM). This work does not study the Observational Model, it only proposes an improvement of the systems polynomial model.

Definition 3. Context Network: A graph formed by all the elements of the context set of the system according to the way proposed by ATMS [4]. In our work this context network will be enriched with the Context Analytical Redundancy Constraints.

Definition 4. Context Analytical Redundancy Constraint (CARC): It is a set of constraints derived from SPM and in such a way that only the observed variables are related. In this work, we are only dealing with the models defined by polynomial equality constraints. In these constraints, their truth value can be evaluated from the system observed variables through the corresponding monitorization.

3 A heat exchangers system: A case study

To explain our methodology, we will apply it to the system shown in Figure 1 and that it was presented in [11]. This system consists of six heat exchangers, with three flows f_i coming in at different temperatures t_i. The functions of the system are described by polynomial constraints, coming from three kinds of balance:

$\sum_i f_i = 0$: mass balance at each node,
$\sum_i f_i * t_i = 0$: thermal balance at each node,
$\sum_{in} f_i * t_i - \sum_{out} f_j * t_j = 0$: enthalpic balance for each heat exchanger.

The system has 34 polynomial equations and 54 variables, from which 28 are observable: t_{11}, t_{12}, t_{13}, t_{16}, t_{17}, t_{18}, t_{19}, t_{112}, t_{21}, t_{26}, t_{27}, t_{212}, t_{31}, t_{33},f_{11}, f_{12}, f_{13}, f_{16}, f_{17}, f_{18}, f_{19}, f_{112}, f_{21}, f_{26}, f_{27}, f_{212}, f_{31} and f_{33}. There is no direct measure of the rest of the variables. This defines three different subsystems, each one formed by two exchangers: {E1, E2}, {E3, E4} and {E5, E6}. Each of the six exchangers and each of the eight nodes of the system are considered as components whose correct functioning must be verified.

For the example presented in Figure 1, the SPM is stored in a CDB table, as we will explain in Section 4.2. The context network for this

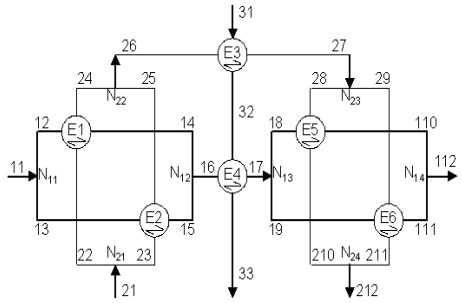

Figure 1. System of heat exchangers

example is too large to be shown, but in order to clarify the concepts, we are including in this paper a subsystem of the system of heat exchanges in Figure 2. This subsystem includes the components $\{N_{12}, N_{21}, N_{22}, E_1, E_2\}$.

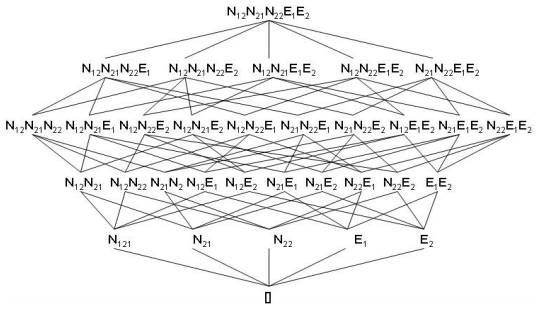

Figure 2. Context Network for a part of the system of heat exchangers

4 Computing possible minimal conflicts

The model which reflects the system structure and behavior is presented by a set of polynomial constraints. The variables of these constraints can be observable or non-observable. All this information is stored in a CDB which relates components, polynomials, variables and contexts. The idea is to produce an equivalent constraints model which has the same solution as the original one, but only with observable variables. In order to produce this model we will use Gröbner bases [1] as symbolic technique.

4.1 Gröbner Bases

Gröbner bases theory is the origin of many symbolic algorithms used to manipulate multiple variable polynomials. It is a generalization of Gauss' elimination of multivariable lineal equations and of Euclides' algorithm for one-variable polynomial equations. Gröbner bases has better computational properties than the original system. We can determine if a system can be solved or not.

The main idea is to transform the polynomial constraint set into a standard form for the resolution of problems. Having the set of equality polynomial constraints of the form $P = 0$, Gröbner bases produce an equivalent system $G = 0$ which has the same solution as the original one, but generally easier to be solved.

For our work, we have a function called GröbnerBasis, which calculates Gröbner bases by means of a finite set of polynomial equations (SPM) and a set of observable and non-observable variables.

This function allows building the context network. The signature of GröbnerBasis function looks like this:

GröbnerBasis({Polynomials},
{Observable Variables},
{Non-observable Variables})

Let us consider, for instance, the context represented by the components $\{N_{12}E1E2\}$. Then, *GröbnerBasis* function takes the parameters:

GröbnerBasis({polynomialsOf($N_{12}, E1, E2$)},
$\{f_{16}, f_{12}, f_{13}, t_{16}, t_{12}, t_{13}\}$,
$\{f_{14}, f_{15}, f_{13}, f_{15}, t_{14}, t_{15}, t_{13}, t_{15}\}$)

The result would be the system of polynomial constraints:

$\{f_{12} + f_{13} - f_{16} = 0\}$

4.2 Constraint Database Architecture

One of the difficulties in diagnosing a system is handling the information, therefore we have important reasons to use CDBs in model-based diagnosis:

1. By using CDBs, it is possible to add or delete some components when our system changes. In this way, rebuilding the full problem is not necessary.
2. If we do not use a CDB and the execution of the algorithm diagnosis fails, while being executed, we must reexecute the full problem because there is not partial information stored.
3. CDBs allow using the power of SQL in order to query the database and obtain the necessary information.

First of all, we are going to explain the database architecture, and how to store the information:

1. **Components:** This table contains the names and identifiers of the components which make up the system.
2. **Polynomials:** This table contains the different behaviors of the components. The components can have more than one polynomial associated with them, like the example of Figure 1. An example of this table for our problem is shown in Table 1.

IdComponent	Constraint
N_{12}	$f_{14} + f_{15} - f_{16}$
N_{12}	$f_{14} * t_{14} + f_{15} * t_{15} - f_{16} * t_{16}$
N_{21}	$f_{21} - f_{22} - f_{23}$
N_{21}	$f_{21} * t_{21} + f_{22} * t_{22} - f_{23} * t_{23}$
N_{22}	$f_{24} - f_{25} - f_{26}$
N_{22}	$f_{24} * t_{24} + f_{25} * t_{25} - f_{26} * t_{26}$
E_1	$f_{12} - f_{14}$
E_1	$f_{22} - f_{24}$
E_1	$f_{12} * t_{12} - f_{14} * t_{14} + f_{22} * t_{22} - f_{24} * t_{24}$
E_2	$f_{13} - f_{15}$
E_2	$f_{23} - f_{25}$
E_2	$f_{13} * t_{13} - f_{15} * t_{15} + f_{23} * t_{23} - f_{25} * t_{25}$

Table 1. Polynomial table

3. **ContextNetwork:** This table represents all the relations that the process must study in order to obtain the minimal possible conflict context. The table has potentially $2^n - 1$ combinations of elements, where n is the number of components that constitute the system.
4. **Variables:** All the variables of the system are stored, observable and non-observable. An example is shown in Table 2.

IdVariable	VarName	Observable
25	t_{212}	Yes
26	t_{31}	Yes
27	t_{33}	Yes
28	f_{14}	No
29	f_{15}	No
30	f_{110}	No

Table 2. Variables Table

5. **VariablePolynomials:** This table represents the variables in each polynomial. This table is important because in order to obtain Gröbner bases we need to send the observable and non-observable variables of the polynomials.
6. **Constraints:** All the constraints are stored in this table. We will fill in this table with the GröbnerBasis results.
7. **ConstraintNet:** This table relates each context to the constraints.

Explanations about CDB tables are shown in Figure 3.

4.3 First Improvement: Reduction Algorithm

If we do not reduce the problem it will be necessary to study $2^{14} - 1$ contexts, and we will obtain 64 constraints. Some of these 64 constraints are redundant because some of them are a lineal combination of others. For that reason, we must detect these combinations and reduce them, as it is done in [6].

In order to improve the time in obtaining the possible minimal conflictive contexts and, therefore, obtain just the necessary information, we propose to use CDB and SQL standard.

Using CDBs and SQL query language, we are proposing a new way to study only the necessary contexts in order to reduce the number of contexts we propose the algorithm of Figure 4. This algorithm offers a way to improve the determination of minimal possible conflict contexts without creating all the contexts nor calling GröbnerBasis function every time.

In order to explain the algorithm, we need to add two new definitions.

Definition 5. *Observable Context*: A context without non-observable variables, that means that we must not eliminate any variable. An example of an observable context is $\{N_{11}\}$.

Definition 6. *Relevant Context*: A context whose components have, at least, one polynomial whose non-observable variables are also in other polynomial of the context. If we call GröbnerBasis function in other cases, we will not obtain any important results, because it is not possible to eliminate all non-observable variables from, at least, one polynomial of all the context's components:

C is a relevant context if
$\mathbf{C} \equiv \bigcup_i \{c_i\} \mid \forall\, c_i \in \mathbf{C} \cdot \exists\, p_i \in c_i$
$\mid \forall\, \mathbf{x} \in \mathbf{NonObsVar}(p_i) \cdot \mathbf{x} \in \mathbf{C}$

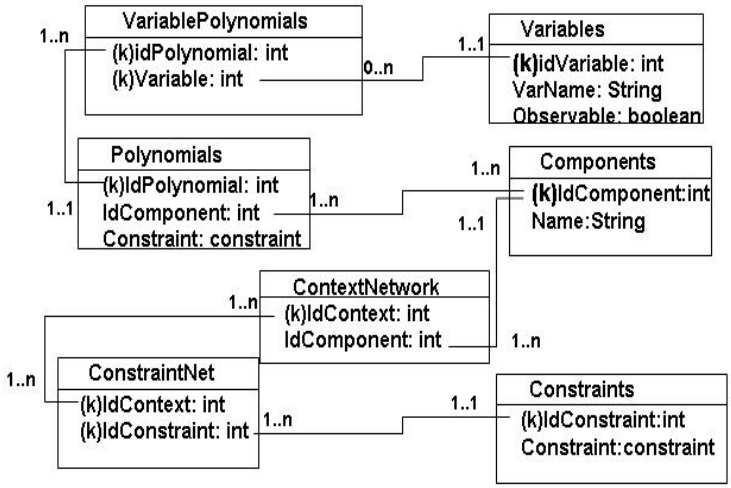

Figure 3. Constraint Database Architecture (k: Primary Key)

```
1 for(i:=1 to i=numComponents)
2    j:=1
3    boolean promising:=true
4    while(promising AND j + i ≤numComponents)
5       if(IsAnObservableContext(i))
6          AddContext(i)
7          UpdateTables(i)
8          promising:=false
9       else
10            Set contexts=ObtainContexs(i, j))
11            for each c of contexts:
12               if (RelevantContext(c))
13                  AddContext(c)
14                  CallGröbner(c)
15               endif
16            endforeach
17       endif
18       j := j + 1
19    endwhile
20 endfor
```

Figure 4. Pseudocode of the reduction algorithm.

where c_i is a component, p_i a polynomial and NonObsVar(p_i)
the set of non-observable variable of p_i

Methods of the reduction algorithm:

- *IsAnObservableContext(Integer i):* This function returns true if the context number i is observable.
- *AddContext(Context c):* This function adds the context *c* to table ContextNetwork.
- *UpdateTables(Context c):* This function stores all the polynomial constraints of the context *c* in tables ConstraintNet and Constraint. Here it is not necessary to call GröbnerBasis because these polynomials do not have non-observable variables.
- *ObtainContext(Integer i, Integer j):*This function returns all the contexts with *j* components which have the component *i* among them.
- *RelevantContext(Context c):* This function returns true if the context *c* is a relevant context.

As we have just explained in this section, it is not necessary to study the full system. But if the system generates all the combinations, the time spent is very high. Moreover, if we try to execute all the contexts, we get redundant information that should be handled later. We propose a way to reduce the use of GröbnerBasis function, only using it when it is necessary to obtain relevant and non-redundant results. Let us see some examples:

Example 1:
Context: N_{22}, E1 and E2
In this case the component E1 does not have any polynomial with all non-observable variable couple with other component.

Example 2:
Context: N_{12}, E1 and E2
In this case all the components have any polynomial with all non-observable variable couple with other component.

In order to implement this idea, we propose an SQL query to know which are the non-observable variables of a *polynomial*, which are also in the same *context* but in different component.

SELECT DISTINCT v.VARNAME
FROM VARIABLES v, VARIABLES v2,
VARIABLEPOLYNOMIALS cv, POLYNOMIALS c,
VARIABLEPOLYNOMIALS cv2, POLYNOMIALS c2,
CONTEXTNETWORK rc, CONTEXTNETWORK rc2,
WHERE c.ID=*polynomial* AND
c.IDCOMPONENT=rc.IDCOMPONENT AND
rc.ID=*context* AND c.ID=cv.ID
AND cv.VARIABLE=v.IDVARIABLE AND
v.OBSERVABLE=false AND c.ID<>c2.ID AND
rc2.ID=rc.ID AND c2.ID=cv2.ID AND
c2.IDCOMPONENT=rc2.IDCOMPONENT AND
cv.VARIABLE=cv2.VARIABLE AND
c.IDCOMPONENT<>c2.IDCOMPONENT

And with the next query, we will know what are the non-observable variables of this polynomial:

SELECT v.VARNAME
FROM VARIABLES v, VARIABLEPOLYNOMIAL cv
WHERE cv.ID=*polynomial* AND
v.IDVARIABLE=cv.VARIABLE AND
v.OBSERVABLE=false

Comparing both results, we will know if all the non-observable variables of a polynomial are in any other components.
With these two queries we can check if it is a relevant context.

- *CallGröbner():* We build GröbnerBasis function call with information from the tables ContextNetwork, Polynomials, Variable-Polinomial and Components. The results will be stored if they are not in the table Constraint. Finally, we will store the constraint and the corresponding context in the table CostraintNet.

The algorithm studies each possible context before being created and calls GröbnerBasis function. At line 1 each component is selected, and with line 4 it will be possible to study all the contexts which have these components with j size. At line 5 the algorithm studies if the context has only one component ($j == 1$), and if it is an observable component. In this case we do not have to study all the possible contexts with the component i, because they will not be relevant. To avoid this useless approach (example 3), we use the boolean variable *promising*.

Example 3:
Context: N_{11}, E3 and E4
Really this context is constituted by two subcontexts: $\{N_{11}\}$ and $\{E3, E4\}$. Thereby it is not necessary to study the full context because we will obtain redundant information

	Without reduction	With reduction
Created contexts	16384	43
Calls to GröbnerBasis	16384	41
Number of constraints	67	17 (Figure 4)
Time	4 days and 2 hours	44 seconds

Table 3. Comparing solutions

In Table 3 is shown how our solution improve the firs approach. With our solution we only call GröbnerBasis function 41 times, because the contexts $\{N_{11}\}$ and $\{N_{13}\}$ are observable contexts.

1	$f_{11} - f_{12} - f_{13}$
2	$f_{11} * t_{11} - f_{12} * t_{12} - f_{13} * t_{13}$
3	$-f_{12} - f_{13} + f_{16}$
4	$f_{21} - f_{26}$
5	$-(f_{13} * t_{12}) + f_{16} * t_{12} + f_{13} * t_{13} - f_{16} * t_{16} + f_{26} * t_{21} - f_{26} * t_{26}$
6	$-f_{112} + f_{18} + f_{19}$
7	$f_{212} - f_{27}$
8	$f_{31} - f_{33}$
9	$f_{26} - f_{27}$
10	$f_{21} - f_{27}$
11	$-(f_{17} * t_{16}) + f_{17} * t_{17} - f_{27} * t_{26} + f_{27} * t_{27} - f_{33} * t_{31} + f_{33} * t_{33}$
12	$f_{16} - f_{17}$
13	$f_{13} * t_{12} - f_{17} * t_{12} - f_{13} * t_{13} + f_{17} * t_{17} - f_{27} * t_{21} + f_{27} * t_{27} - f_{33} * t_{31} + f_{33} * t_{33}$
14	$-f_{12} - f_{13} + f_{17}$
15	$f_{18} * t_{112} + f_{19} * t_{112} - f_{18} * t_{18} - f_{19} * t_{19} + f_{27} * t_{212} - f_{27} * t_{27}$
16	$f_{17} - f_{18} - f_{19}$
17	$f_{17} * t_{17} - f_{18} * t_{18} - f_{19} * t_{19}$

Table 4. Minimal conflict constraints for the system (CARCs)

4.4 Second Improvement: Elimination redundant contexts

The key idea is to eliminate the contexts whose children in the context network have the same constraints as the father. An example of this redundance is shown in Figure 5. In this case we can eliminate the context $\{N_{14}, E3, E4, E5, E6\}$, because all their constraints also are in the children of this context.

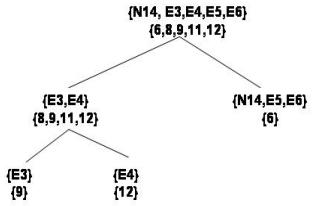

Figure 5. Elimination of Redundancies

With the techniques of eliminating contexts with redundant constraints, we have deleted 31 contexts, thereby we have 12 contexts and 17 constraints, as we see in Figure 6.a. This improvement is done using a module to eliminate some linear combinations.

To detect more redundancies we will study each tree of context using Gröbner Bases too. For example, to study the redundancies of the tree marked with the discontinuous line in the Figure 6.a, we have to check whether the constraints of the root tree has redundant information. To do it, we build a GröbnerBasis function call as:

GröbnerBasis[{CARCs-(8,9,10,11,12,13,14)},
{f12, f13, f16, f26, f31, f33, f27, f17, f21,
t12, t13, t16, t31, t33, t27, t17, t21}, {}]
And other with the rest of the constraint tree:
GröbnerBasis[{CARCs-(3,4,5,8,9,11,12)},
{f12, f13, f16, f26, f31, f33, f27, f17, f21, t12, t13, t16, t31,

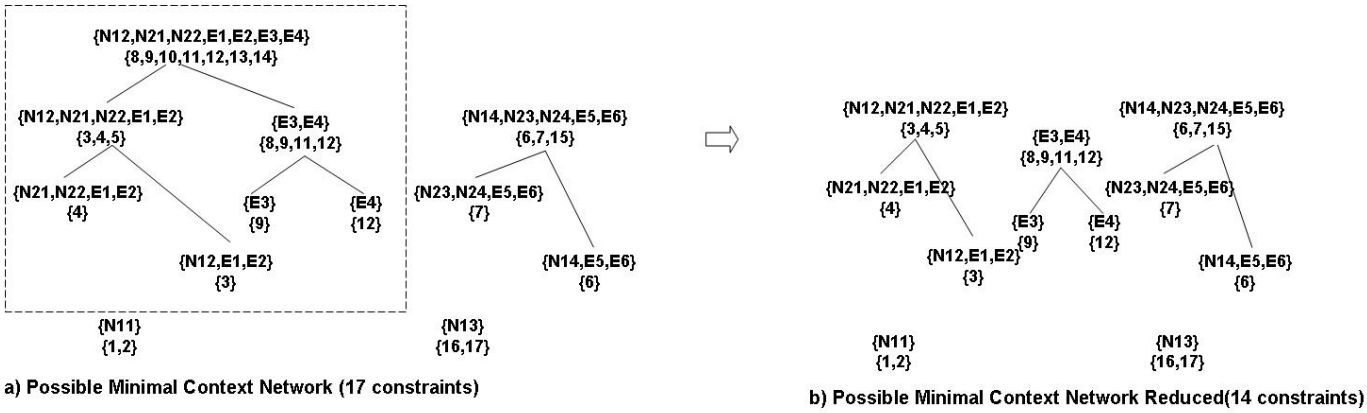

Figure 6. Step to obtain the Possible Minimal Context Network for the system

t33, t27, t17, t21}, {}]
In both cases we obtain:
{-f17 t16 + f17 t17 - f21 t26 + f21 t27 - f33 t31 + f33 t33,
f21 - f27, f31 - f33, f21 - f26, f16 - f17,
-f13 t12 + f17 t12 + f13 t13 - f17 t16 + f21 t21 - f21 t26,
-f12 - f13 + f17}

It means that whenever there are conflicts in some children of the tree, there are also conflicts in the root, therefore the information of the father is redundant because it does not improve the fault detection. In our example we should eliminate the constraints number 10, 13 and 14, obtaining a new minimal context network shown in Figure 6.b.

The final set of CARCs obtained is the same of the Table 4, but without constraints 10, 13 and 14.

All these ideas have been implemented using SQL and a Java(TM) program.

5 Conclusions and future works

In this paper we propose a CDB architecture to store polynomial constraints using standard SQL and Java^{TM} language to obtain and handle the constraint information. One of the most important advantages of this paper is the computational improvement in the calculation of the possible minimal set conflicts. Another important advantage is the power of SQL offered to store and get information in CDBs automatically.

As future works we want to improve our methodology dividing the system into several subsystems. It is interesting in systems whose observable variables allow to know the problem by parts. The problem of the heat exchangers is an example of this. Also, we are considering to study how the minimal context network changes when some polynomials change, and to look for techniques to avoid restudying all the system.

6 Acknowledgements

This work has been funded by the Ministerio de Ciencia y Tecnologa of Spain (DPI2003-07146-C02-01) and the European Regional Development Fund (ERDF/FEDER).

References

[1] Buchberger B. Gröbner bases An algorithmic method in polynomial ideal theory. In Multidimensional Systems Theory, N. K. Bose, ed., D. Reidel Publishing Co., pag 184-232, 1985.
[2] Davis R. Diagnostic reasoning based on structure and behavior In Artificial Intelligence, 24 pag 347-410, 1984.
[3] De Kleer J., Mackworth A., and Reiter R. Characterizing diagnoses and systems. En Artificial Intelligence, 56(2-3), 197-222, 1992.
[4] De Kleer J. An Assumption-based Truth Maintenance System. Artificial Intelligence 28(2) pp 127-161, 1986
[5] Garcia de la Banda M., Stuckey P., Wazny J. Finding all minimal unsatisfiable subsets Proc. Of the 5th ACM Sigplan Internacional 2003.
[6] Gasca R.M, Del Valle C, Ceballos R. and Toro M. An Integration of FDI and DX approaches to polynomial models DX 2003
[7] Gasca R.M., Ortega J.A., Toro M. and De la Rosa F Diagnosis dirigida por restricciones simbólicas para modelos polinmicos. Terceras Jornadas de trabajo sobre Metodologias Cualitativas Aplicadas a los Sistemas Socioeconomicos, Valladolid Julio 2001.
[8] Goldin, D.Q. Constraint Query Algebras My doctorate thesis, Jan. 1997
[9] Goldin, D.Q., Kanellakis, P.C. Constraint Query Algebras Constraints Journal E. Freuder editor, 1st issue, 1996.
[10] Grumbach S., Rigaux P., Scholl M., and Segoun L. Spatial constraint database In S. Cluet and R. Hull, editors,Proceedings of Database Programming Languages (DBPL'97), volume 1369 of Lecture Notes in Computer Science, pages 38-59. Springer-Verlag, 1998.
[11] Guernez C. et al Fault detection and isolation on non linear polynomial system. 15th IMACS Word Congress on Scientific, Computation, Modelling and Applied Mathematics, 1997.
[12] Helm R., Marriott K. and Odersky M. Constraint-based query optimization for spatial databases. In Tenth ACM Symposium on the Principles of Database Systems, pages 181-191, Denver, CO, May 1991.
[13] Kanellakis P.C., Kuper G.M. and Revesz P.Z. Constraint Query Languages. Symposium on Principles of Database Systems, pp. 299-313. 1990.
[14] Kuper G., Libkin L. and Paredaes J.: Constraint Databases. Springer, 1998.
[15] Pulido J. B. Posibles conflictos como alternativa al registro de dependencias en lnea para el diagnstico de sistemas continuos. Tesis Doctoral de la Universidad de Valladolid, 2000.
[16] Reiter R. A theory of diagnosis from first principles. In Artificial Intelligence, 32(1), pag 57-96, 1987.
[17] Revesz P.: Introduction to Constraint Databases. ISBN:0-387-98729-0 Springer, 2001.
[18] Strumptner M, Wotawa F. Diagnosing Tree Structured System IJCAI pag 440-445, 1997