Applying Constraint Databases in the Determination of Potential Minimal Conflicts to Polynomial Model-Based Diagnosis

Maria Teresa Gómez López, Rafael Ceballos Guerrero, Rafael Martínez Gasca, and Carmelo del Valle Sevilla

Departamento de Lenguajes y Sistemas Informáticos,
Escuela Técnica Superior de Ingeniería Informática, Universidad de Sevilla, Spain
{mayte,ceballos,gasca,carmelo}@lsi.us.es

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 may be represented by constraints associated to the components. The variables of these constraints can be observable or non-observable, depending on the situation of the sensors. In order to obtain the potential minimal diagnosis in a system, an important issue is related to finding out the potential minimal conflicts in an efficient way. We consider that Constraint Databases represent an excellent option in order to solve this problem in complex systems.
In this work we have used a novel logical architecture of Constraint Databases which has allowed obtaining these potential conflicts by means of the corresponding queries. Moreover, we have considered Gröbner Bases as a projection operator to obtain the potential minimal conflicts of a system. The first results obtained on this work, which are shown in a heat exchangers example, have been very promising.

1 Introduction

In the 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 that these faults can produce a negative impact on the environment, what has to be avoided. For this reason, in order to keep the systems within the desired security, production and reliability levels, mechanisms which allow the detection and diagnosis of the faults produced in the systems have to be developed.

Diagnosis allows us to identify the failures in a system. Most approaches proposed in the last decade use models (FDI and DX approaches) based on the knowledge of the system to diagnose. These models can be formally well structured or can be known by means of an expert's experience and data of the system or process. Sometimes, a combination of both types of knowledge can also be presented. In the area of DX, the first work related to diagnosis was presented with the objective of identifying faults in the component systems based on the structure and its behavior [3]. The first implementations to perform diagnosis were DART [11] and GDE [5], which used different inference mechanisms to detect possible faults.

Diagnosis formalization was presented in [20, 6], where a general theory was proposed for the problem of explaining the discrepancies between the observed and correct behaviors that the mechanisms subject to the diagnosis process (logical-based diagnosis) have. Based on this, most DX approaches for components characterize the diagnosis of a system as a collection of potential minimal sets of failing components which explain the observed behaviors (symptoms). The importance of having a good model in DX can be deduced from this. This kind of diagnosis can correctly diagnose the important parts of the component systems that are failing. An exhaustive revision of the approaches about diagnosis task automation can be found in [7], and a discussion about model-based diagnosis for applications can be consulted in [2].

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 gets data from several observations, equations of the system description and contexts at an enormous rate. In engineering applications, it is often overlooked the storage of these data and query processing.

In this paper, we present how some issues concerning Constraint Database can be the improvement of the efficiency in some phases of the model-based diagnosis. In concrete we tackle the determination of potential minimal conflicts of a system. A conflict is a set of assumptions where at least one 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. In this approach, conflicts are identified in the constraint propagation process 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 main disadvantage of using it is a large number of possible conflicts 2^n, where n is the number of components. In the previous years, this problem has been an active area of research, in order to find a minimal conflict [18] or the finding all potential minimal conflicts, using CS-Tree [15] or using preprocessing, independence and incrementally features of initial model [9]. In the DX community there are a lot of works in this sense as the calculation the minimal chains which can be evaluated, the minimal models which can also be evaluated [19] and symbolic processing algorithms (Gröbner bases) of the initial model [10]. The use of Gröbner bases in the FDI community were also proposed in previous works [8].

Our work presents a novel methodology that uses Constraint Databases technology for the determination of potential minimal conflicts with the object of promoting the advantages of this technology and its uses in real problem of industrial diagnosis. There are some positive features to many model-based applications that can be exploited by a Constraint Database. First the easy representation of the component-based model of a technical system by means of a set of algebraic constraints and during the search in the space of possible contexts to model-based diagnosis, the constraints of the different contexts differ in only a few constraints. This similarity can be exploited by incremental solving techniques.

In our paper, the model is stored in a Polynomial Constraint Database and a preprocessing step reduces in a significant way the number of possible contexts to treat by means of symbolic techniques. This technique is based on the algorithm for computing Gröbner bases given by Buchberger in [1]. This algorithm is our projection operator and we will use it to eliminate the non-observable variables of the constraints in the different contexts.

Moreover the relational model is not able to represent the scientific and engineering data. For this reason we consider Constraint Databases [17, 16, 21, 12, 13] as a good tool for our reasoning processing.

Our paper has been organized as follows: firstly, we present definitions and notation which allow us to formalize the subsequent operations. In sections 3 we show an example to prove our solution and use a Constraint Database architecture to treat the information. Afterwards, we show the obtained results when we applied our methodology to the set of heat exchangers system example. Finally we present our conclusions and future works in this research line.

2 Definitions and Notation

The definitions and notation used are based on the concepts developed in the diagnosis community, based itself on the logic (DX) and on redundancies (FDI). The objective is that the synergy of both approaches will produce diagnosis results which are as representative as possible of what is happening in the system in the shortest time. As it has already been mentioned, model-based diagnosis requires a system model. This time, we will only deal with the case which has a model of system constraints that derives from its own structure, and which has links between components (structural model) and the behavior of each model component. With this model and with the idea of formalizing the diagnosis process, the definitions and notation used in the development of this work need to be exposed.

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. Then, the following tuple for a system polynomial model is obtained SPM (P,Vob,Vnob).

Definition 2. Diagnosis Problem (DP): It can be defined by means of a tuple formed by a System Polynomial Model 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 reflects, in a minimal way, the information of the possible failing components DP(SPM,OM). In this work we are not going to study the Observational Model, we only propose an improvement of the systems polynomial model.

Definition 3. Context Network: It is 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): 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 observed variables of the system through the corresponding monitorization.

3 A Heat Exchangers System: A Case Study

To explain our methodology, we have been applied it to a system, like the one shown in figure 1 from [10] and [14]. In this system, consisting of six heat exchangers, three flows f_i come in at different temperatures t_i. The behaviors of the system is described by polynomial constraints coming from three different kinds of balances:

$\sum_i f_i = 0$: 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.

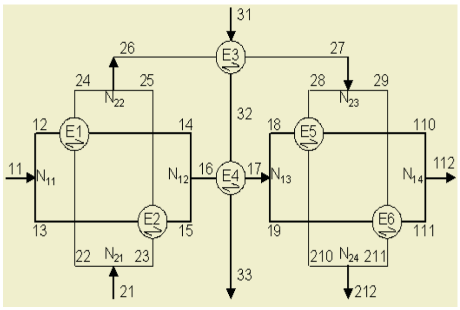

Fig. 1. System of heat exchangers

The resulting system, thus, consists of 34 equations and 54 variables, from which 28 are observable: 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}, f_{33}, 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} and t_{33}. There is no a 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 have to be verified.

Table 1. System Polynomial Model of the System of Heat Exchangers

C.	*Constraints*	*C.*	*Constraints*
N_{11}	$\boldsymbol{f}_{11}$-$\boldsymbol{f}_{12}$-$\boldsymbol{f}_{13}$	E_1	$\boldsymbol{f}_{12}$-f_{14}
	$\boldsymbol{f}_{11}\cdot\boldsymbol{t}_{11}$-$\boldsymbol{f}_{12}\cdot\boldsymbol{t}_{12}$-$\boldsymbol{f}_{13}\cdot\boldsymbol{t}_{13}$		f_{22}-f_{24}
N_{12}	$f_{14}+f_{15}$-$\boldsymbol{f}_{16}$		$\boldsymbol{f}_{12}\cdot\boldsymbol{t}_{12}$-$f_{14}\cdot t_{14}+f_{22}\cdot t_{22}$-$f_{24}\cdot t_{24}$
	$f_{14}\cdot t_{14}+f_{15}\cdot t_{15}$-$\boldsymbol{f}_{16}\cdot\boldsymbol{t}_{16}$	E_2	$\boldsymbol{f}_{13}$-f_{15}
N_{13}	$\boldsymbol{f}_{17}$-$\boldsymbol{f}_{18}$-$\boldsymbol{f}_{19}$		f_{23}-f_{25}
	$\boldsymbol{f}_{17}\cdot\boldsymbol{t}_{17}$-$\boldsymbol{f}_{18}\cdot\boldsymbol{t}_{18}$-$\boldsymbol{f}_{19}\cdot\boldsymbol{t}_{19}$		$\boldsymbol{f}_{13}\cdot\boldsymbol{t}_{13}$-$f_{15}\cdot t_{15}+f_{23}\cdot t_{23}$-$f_{25}\cdot t_{25}$
N_{14}	$f_{110}+f_{111}$-$\boldsymbol{f}_{112}$	E_3	$\boldsymbol{f}_{26}$-$\boldsymbol{f}_{27}$
	$f_{110}\cdot t_{110}+f_{111}\cdot t_{111}$-$\boldsymbol{f}_{112}\cdot\boldsymbol{t}_{112}$		$\boldsymbol{f}_{31}$-f_{32}
N_{21}	$\boldsymbol{f}_{21}$-f_{22}-f_{23}		$\boldsymbol{f}_{26}\cdot\boldsymbol{t}_{26}$-$\boldsymbol{f}_{27}\cdot\boldsymbol{t}_{27}+\boldsymbol{f}_{31}\cdot\boldsymbol{t}_{31}$-$f_{32}\cdot t_{32}$
	$\boldsymbol{f}_{21}\cdot\boldsymbol{t}_{21}$-$f_{22}\cdot t_{22}$-$f_{23}\cdot t_{23}$	E_4	$\boldsymbol{f}_{16}$-$\boldsymbol{f}_{17}$
N_{22}	$f_{24}+f_{25}$-$\boldsymbol{f}_{26}$		f_{32}-$\boldsymbol{f}_{33}$
	$f_{24}\cdot t_{24}+f_{25}\cdot t_{25}$-$\boldsymbol{f}_{26}\cdot\boldsymbol{t}_{26}$		$\boldsymbol{f}_{16}\cdot\boldsymbol{t}_{16}$-$\boldsymbol{f}_{17}\cdot\boldsymbol{t}_{17}+f_{32}\cdot t_{32}$-$\boldsymbol{f}_{33}\cdot\boldsymbol{t}_{33}$
N_{23}	f_{27}-f_{28}-f_{29}	E_5	f_{18}-f_{110}
	$f_{27}\cdot\boldsymbol{t}_{27}$-$f_{28}\cdot t_{28}$-$f_{29}\cdot t_{29}$		f_{28}-f_{210}
N_{24}	$f_{210}+f_{211}$-$\boldsymbol{f}_{212}$		$\boldsymbol{f}_{18}\cdot\boldsymbol{t}_{18}$-$f_{110}\cdot t_{110}+f_{28}\cdot t_{28}$-$f_{210}\cdot t_{210}$
	$f_{210}\cdot t_{210}+f_{211}\cdot t_{211}$-$\boldsymbol{f}_{212}\cdot\boldsymbol{t}_{212}$	E_6	$\boldsymbol{f}_{19}$-f_{111}
			f_{29}-f_{211}
			$\boldsymbol{f}_{19}\cdot\boldsymbol{t}_{19}$-$f_{111}\cdot t_{111}+f_{29}\cdot t_{29}$-$f_{211}\cdot t_{211}$

V_{ob}=f_{11},f_{12},f_{13},f_{16},f_{17},f_{18},f_{19},f_{21},f_{26},f_{27},f_{112},f_{212},f_{31},f_{33},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}

V_{nob}=f_{14},f_{15},f_{21},f_{22},f_{23},f_{24},f_{25},f_{28},f_{29},f_{210},f_{211},f_{110},f_{111},f_{32},t_{14},t_{15},t_{110},t_{111},t_{22}, t_{23},t_{24},t_{25},t_{28},t_{29},t_{210},t_{211},t_{32}

For the example presented in figure 1, the SPM is represented in Table 1. The context network for this example is too large to be shown, but in order to clarify this concept, we present only one subsystem in the figure 2. This subsystem includes the components $\{N_{12}, N_{21}, N_{22}, E_1, E_2\}$.

4 Computing Potential Minimal Conflicts

The model which reflects the system structure and behavior contains the constraints that link the system inputs and outputs. Many times some intermediate variables are not observable and do not allow to determine whether there are faults in the components in a direct way. Then our idea is to produce an equivalent constraints model which has the same solution as the original one, but without non-observable variables. In order to transform the polynomial constraints set of the system, we apply a function that obtains Gröbner bases.

To study the potential minimal conflicts with common techniques, we must rebuild the full problem, if our system changes because we add or delete some components. To solve this problem we will store information in a Constraint Database. Also, if we do not use a Constraint Database and the execution of the program diagnosis fails while being executed, we must reexecute the full problem. It is because there is not any partial information stored. If we save the

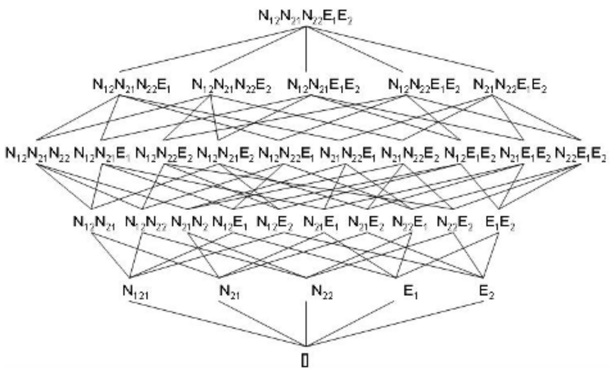
N12N21N22E1E2
N12N21N22E1 N12N21N22E2 N12N21E1E2 N12N22E1E2 N21N22E1E2
N12N21N22 N12N21E1 N12N22E2 N12N21E2 N12N22E1 N21N22E1 N21N22E2 N12E1E2 N21E1E2 N22E1E2
N12N21 N12N22 N21N22 N12E1 N12E2 N21E1 N21E2 N22E1 N22E2 E1E2
N121 N21 N22 E1 E2
[]

Fig. 2. Context Network for a subsystem of the heat exchangers example. The components included are $\{N_{12}, N_{21}, N_{22}, E_1, E_2\}$

information in a database step by step, we can continue the process with the obtained information. Constraint Databases allows us to use the power of SQL standard in order to query the database and to obtain the necessary information.

In the following two subsections we explain the most important properties about Gröbner bases and the Constraint Database architecture. In the last section we propose an algorithm to obtain the potential minimal conflicts based on Gröbner bases and Constraint Databases.

4.1 Gröbner Bases

Gröbner bases theory is the origin of many symbolic algorithms used to manipulate multiple variable polynomials. The algorithm used for the polynomial equations system is based on the ideas proposed in [1]. 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 be said that it is very easy to determine if the 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 it is generally easier to solve.

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

This function obtains the different contexts of a particular model and allows the building of 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 following 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 following 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. In this paper we propose to store the information in a relational database. It will allow us to store partial results.
Also, it is important to highlight the power of the query language in a database. Using a database helps to improve the diagnosis and to access data.

First of all, we are going to explain the database architecture, and how we store the information. It is shown in figure 3. The semantics of these storage elements are explaining in the following items:

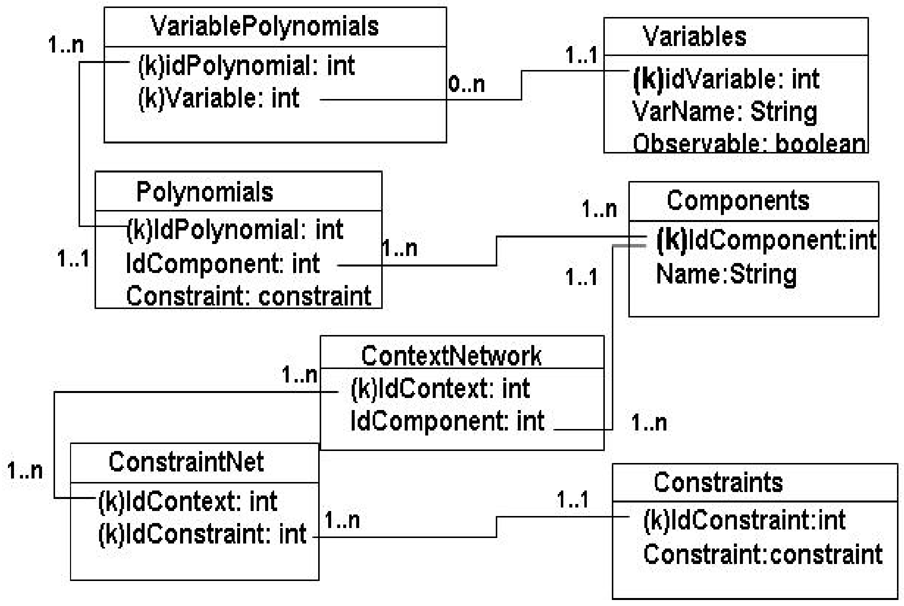

Fig. 3. Constraint Database Architecture (k: Primary Key)

Explanations about Constraint Database tables (figure):

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 the figure 1. An example of this table for our problem is shown in table 2.

Table 2. Polynomial table

IdPolynomial	IdComponent	Constraint
0	0	$f_{11} - f_{12} - f_{13}$
1	0	$f_{11} * t_{11} - f_{12} * t_{12} - f_{13} * t_{13}$
2	1	$f_{14} + f_{15} - f_{16}$
3	1	$f_{14} * t_{14} + f_{15} * t_{15} - f_{16} * t_{16}$

3. **ContextNetwork:** This table represents all the relations that the process must study to obtain the minimal possible conflict context. The table has potentially 2^n combination of elements, where n is the number of components that constitute the system.
4. **Variables:** Here all the variables which take part in the system are stored, observable as well as non-observable. An example of it is shown in the table 3.

Table 3. Variables Table

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

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

4.3 Constraint Databases Approach

If we call GröbnerBasis for all contexts (2^{14} times) to obtain the conflictive contexts will take 4 days and 2 hours, and it will obtain 64 constraints. Some of these 64 constraints are redundant because some of them are linear combination of others. For that reason, we must detect these combinations and reduce them, as it is done in [10].

We propose to use Constraint Databases and the power of SQL standard to query the database and obtain the data. It helps us to improve the time of obtaining the possible minimal conflictive contexts, and therefore obtain just the necessary information.

Using the constraints databases and the SQL query language, we are proposing a new way to improve the first solution. To improve the solution mentioned above, we have avoided these two disadvantages of it. The disadvantages are the handling of the data and the great amount of calls to GröbnerBasis function. To improve it we use the algorithm of the figure 4.

```
for($i$ := 1 to i=numComponents)
  $j := 1$
  boolean promising:=true
  while(promising AND $j + i \leq$numComponents)
    if(IsAnObservableContext($i$))
      AddContext($i$)
      UpdateTables($i$)
      promising:=false
    else
      Set contexts=ObtainContexs($i, j$))
      for each c of contexts:
        if (RelevantContext($c$))
          AddContext($c$)
          CallGröbner($c$)
        endif
      endforeach
    endif
    $j := j + 1$
  endwhile
endfor
```

Fig. 4. Pseudocode of the reduction algorithm

This algorithm offers a way to improve the obtaining of minimal possible conflictive contexts without creating all the contexts nor calling to GröbnerBasis function all the times.

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

Definition 5. *Observable Context*: It is a context without non-observable variables. It means that we do not have to eliminate any variables. An example of an observable context is $\{N_{11}\}$.

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

C is a relevant context if
C$\equiv \bigcup_i \{c_i\} \mid \forall\ c_i \in \mathbf{C} \cdot \exists\ p_i \in c_i$
$\mid \forall\ \mathbf{x} \in$ **NonObsVar**$(p_i) \cdot \mathbf{x} \in \mathbf{C} \setminus c_i$
c_i **being 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():*This method returns true if the context has only one component and this component does not have any polynomial with non-observable variables.
- *AddContext(Context c):*Add to table ContextNetwork the context c.
- *UpdateTables(Context c):*Input all the polynomial constraints of the context c, in the tables ConstraintNet and Constraint. Here it is not necessary to call GröbnerBasis because the polynomials do not have non-observable variables, and Gröbner method does not have any non-observable variables to eliminate.
- *ObtainContext(Integer i, Integer j):*This method returns all the contexts with j components which have the component i in them.
- *RelevantContext(Context c):* Returns true if the context c is a relevant context.

  In section 4.3 it is explained how many calls to Gröbner are necessary to study the full system. But if the system studies all the combinations, the cost of time is very high. Moreover, if we try to execute all the contexts, we will get redundant information that should be handled later. We propose a way to reduce the use of Gröbner, and only use it when it is necessary, when the results are interesting and non-redundant.

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 its non-observable variable couple with other component.

To implement previous idea, we propose an interesting query to know which are the non-observable variables of a *polynomial constraint* which are also in the same *context* but in distinct component.

```
SELECT DISTINCT v.VARNAME
FROM VARIABLES v, VARIABLES v2, VARIABLEPOLYNOMIALS cv,
    VARIABLEPOLYNOMIALS cv2, CONTEXTNETWORK rc,
    CONTEXTNETWORK rc2, POLYNOMIALS c, POLYNOMIALS c2
```

```
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 AND
    rc2.ID=rc.ID c2.IDCOMPONENT=rc2.IDCOMPONENT AND
    c2.ID=cv2.ID AND cv.VARIABLE=cv2.VARIABLE AND
    c.IDCOMPONENT<>c2.IDCOMPONENT
```

And with the next query, we will know which are the non-observable variables of a polynomial constraint:

```
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 another component.

With these two queries we can check if it is a relevant context.

– *CallGröbner():*We build GröbnerBasis function with information of the tables ContextNetwork, Polynomials, VariablePolynomial and Components. After the execution of the function we will pick the results up. If the solution is not in Constraint table, we will add it. Finally, we will store the constraint and the context related in the table CostraintNet.

Explanation of the reduction algorithm:
Our reduction algorithm studies each possible context before it is created and calls GröbnerBasis. In line 1 each component is selected, and in line 4 it will be possible to study all the contexts which have the component i in their j components. In 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 computation (example 3), we use the boolean variable *promising*. Let us see the following example:

Example 3:

Context: N_{11}, E3 and E4

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

With this algorithm, we only create 43 contexts as an alternative to 2^{14} of the first solution. With our solution we only call GröbnerBasis 41 times, because the contexts $\{N_{11}\}$ and $\{N_{13}\}$ are observable contexts. Our reduction algorithm spends 12 minutes and 27 seconds and obtains 17 CARCs which are shown on table 4.

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

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	$f_{17} * t_{17} - f_{18} * t_{18} - f_{19} * t_{19}$

4.4 Context Analytical Redundancy Constraint

In this section we will study, with the help of SQL and Java^{TM} languages, if there are any contexts whose polynomial constraints have already been in another context with less components. One example is shown in the figure 5. In this case we can eliminate the context $\{N_{14}, E3, E4, E5, E6\}$, because all their constraints are in contexts with less components.

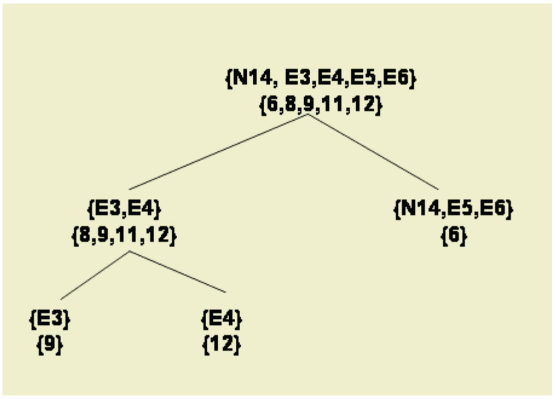

Fig. 5. Elimination of Redundancies

With this elimination technique, we have deleted 31 contexts with redundant constraints. Thereby we have 12 contexts and 17 constraints, as we show in the figure 6. To improve this result we are studying to use a module to eliminate some linear combination.

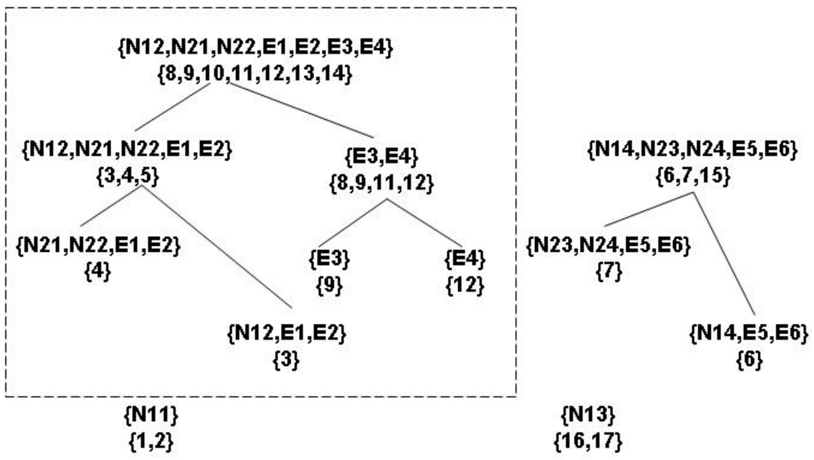

Fig. 6. Minimal Context Network for the system

5 Conclusions and Future Works

In this paper we have proposed a database architecture to store polynomial constraints in a Constraints Database. Using SQL standard and Java languages is possible to obtain and handle the polynomial constraint of the model.The proposed reduction algorithm has several advantages. One is the computational improvement in the calculation of the potential minimal set conflicts. Another advantage is to make the process automatic. Finally it is important highlight the power of SQL that offers to store and get information in Constraint Databases in a easy way.

As future works we want to improve our methodology dividing the system into several subsystems. If the system is divided, it can be studied separately. It is interesting in systems which have observable variables which allow to know the problem by parts. The problem of the heat exchangers is an example of this because it can be divided into five different parts. Also, we are considering to study how the minimal context network changes when some polynomial constraints change, and to look for techniques to avoid restudying all the system.

Anyway, there is a great field to study the creation of systems with components located in different Constraint Databases.

Acknowledgements

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

References

1. B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. *In Multidimensional Systems Theory, N. K. Bose, ed., D. Reidel Publishing Co.*, pages 184–232, 1985.
2. L. Console and O. Dressler. Model-based diagnosis in the real world: Lessons learned and challenges remaining. *In Proceedings IJCAI'99*, pages 1393–1400, 1999.
3. R. Davis. Diagnostic reasoning based on structure and behavior. *In Artificial Intelligence*, 24:347–410, 1984.
4. J. de Kleer. An assumption-based truth maintenance system. *Artificial Intelligence 28(2)*, pages 127–161, 1986.
5. J. de Kleer and Williams B.C. Diagnosing multiple faults. *Artificial Intelligence*, 1987.
6. J. de Kleer, A. Mackworth, and R. Reiter. Characterizing diagnoses and systems. *In Artificial Intelligence*, 56(2-3):197–222, 1992.
7. E. Frisk. The consistency-based approach to automated diagnosis of devices. *In Brewka(ed). Principles of Knowledge Representation, CSLI*, 1996.
8. E. Frisk. Residual generator design for non-linear, polynomial systems - a gröbner basis approach. *In Proc. IFAC Safeprocess 2000, Budapest Hungary*, pages 979–984, 2000.
9. M. Garcia de la Banda, P. Stuckey, and J. Wazny. Finding all minimal unsatisfiable subsets proc. *Of the 5th ACM Sigplan Internacional*, 2003.
10. R.M Gasca, C Del Valle, R. Ceballos, and M. Toro. An integration of fdi and dx approaches to polynomial models. *DX*, 2003.
11. M. Genesereth. The use of design descriptions in automated diagnosis. *Artificial Intelligence*, 24:411–436, 1984.
12. D.Q. Goldin. Constraint query algebras. *Doctorate thesis*, 1997.
13. D.Q. Goldin and P.C. Kanellakis. Constraint query languages. *Constraint Query Algebras Constraints Journal*, 1996.
14. C. Guernez. Fault detection and isolation on non linear polynomial system. *5th IMACS Word Congress on Scientific, Computation, Modelling and Applied Mathematics*, 1997.
15. B. Han and S. Lee. Deriving minimal conflict sets by cs-trees with mark set in diagnosis from first principles. *IEEE Transcations on Systems, man and cybernetics*, 29(2), 1999.
16. P. C. Kanellakis, G. M. Kuper, and P.Z. Revesz. Constraint query languages. *Symposium on Principles of Database Systems*, pages 299–313, 1990.
17. G. Kuper, L. Libkin, and J. Paredaes. *Constraint Databases*. Springer, 1998.
18. Mauss and Mugur Tatar. Computing minimal conflicts for rich constraint languages jakob. *DX*, 2002.
19. J. B. Pulido. Posibles conflictos como alternativa al registro de dependencias en lnea para el diagnstico de sistemas continuos. *Ph. Doctoral Dissertation. Universidad de Valladolid*, 2000.
20. R. Reiter. A theory of diagnosis from first principles. *In Artificial Intelligence*, 32(1):57–96, 1987.
21. P. Revesz. *Introduction to Constraint Databases*. Springer, 2002.