Distributed Model-Based Diagnosis using Object-Relational Constraint Databases

M. T. Gómez-López, R. M. Gasca, C. del Valle and S. Pozo
Departamento de Lenguajes y Sistemas Informticos
University of Seville, Spain
{mayte,gasca,carmelo,sergio}@lsi.us.es

Abstract

This work presents a proposal to diagnose distributed systems utilizing model-based diagnosis using distributed databases. In order to improve aspects as versatility, persistence, easy composition and efficiency in the diagnosis process we use an Object Relational Constraint Database (ORCDB). Thereby we define a distributed architecture to store the behaviour of components as constraints in a relational database to diagnose a distributed system. This work proposes an algorithm to detect which components fail when their information is distributed in several databases, and all the information is not available in a global way. It is also offered a proposal to define, in execution time, the allocation of the sensors in a distributed system.

1 Introduction

Diagnosis allows to determine why a system correctly designed does not work as it was expected. It is based on the monitorization of a system. The diagnosis aim is to detect and to identify the reason of the unexpected behaviour, or in other words, to identify the parts which fail in a system. Our proposal is based on DX community approaches [10, 7]. These works were proposed to find out the discrepancies between the observed and correct behaviors of a system.

In this work we want to extend three important aspects in model-based diagnosis, as how to store the information related to diagnosis process, offer the capacity to study different sensors allocation, and improve the distributed model-based diagnosis where a part of the information is public, and another is private.

Related to the first proposal, how to store the information related to diagnosis process, we use Constraint Databases (CDBs) [8]. CDBs area was developed to store in a database continuous information, this is the case of diagnosis, because it is necessary to store the behaviour and the relation between the components that form a system. In engineering applications it is often overlooked the storage of these data and query processing. For these reasons, we store all the diagnosis information in an ORCDB (Object Relational Constraint Database), where the constraints that represent the components' behaviour are stored as objects in a Relational Database as it is explained in [4].

Most important aspect in our contribution consists on diagnose a system when the information is structurally distributed in different ORCDBs. For example, if we have a system whose components are stored in several databases although the diagnosis has to be analysed in a global way.

Distributed databases have several aspects that have to be analysed, as allocation of data, local database design or algorithm to combine the information [9]. These characteristics are studied in this work and we propose a solution.

This work is organised as follows: Section 2 analyses other previous works in model-based diagnosis area. Section 3 presents an example to explain our proposal and how it is stored in an ORCBD. Section 4 shows the distributed architecture and an example of study. Section 5 presents a distributed algorithm using the previous example. Finally, some conclusions and future work are presented.

2 Model-Base Diagnosis

Fault detection and identification of faulty components are very important from the strategic point of view of companies, due to the economic demands and required environment conservation to remain in competitive markets.

Diagnosis allows us the identification of failures in a system, and with CDBs technology it is possible to make persistent the information and models. Our proposal is based on DX ([10] and [7]). These works were proposed to find out the discrepancies between the observed and correct behaviours of a system.

In this work, a new approach is proposed in order to querying distributed ORCDB obtaining equivalent systems

1550-445X/06 $20.00 © 2006 IEEE

which can be diagnosed. The location of sensors defines which variables are observable, the rest of variables will be non-observable variables. In order to present our proposal in a clear way, some definitions are necessary:

Definition 1. The System Polynomial Model (SPM): It can be defined as a finite set of polynomial equality constraints *P* which determine the system behaviour. This is done by means of the relation between the system non-observable variables and the observable variables which are directly obtained from sensors.

Definition 2. *Context Set (CS)*: Any subset of components which composes the system. There are $2^{ncomp} - 1$ possible context sets, where *ncomp* is the number of components.

Definition 3. *Context Network (CN)*: A graph formed by all the elements of the Context Set of the system according to the way proposed by ATMS [6].

Definition 4. *Context Analytical Redundancy Constraint (CARC)*: A constraint derived from the system, in such a way that only the observable variables are related.

Definition 5. *Observational Model (OM)*: A set of values for the observable variables of the system.

Definition 6. *Possible Minimal Conflict Contexts (PMCC)*: Contexts of the CN which have one or more than one CARC, generated with the elements of the contexts, and which may be checked the consistency of the CARCs for an OM.

Definition 7. *Minimal Conflict Context (MCC)*: A minimal conflict context whose CARCs are unsatisfiable for an OM. The MCCs help us to find out which components are failing.

Definition 8. *Hitting Set (HS)* for a collection of sets $\mathcal{C}$ is a set $\mathcal{H} \subseteq \bigcup_{\mathcal{S} \in \mathcal{C}} \mathcal{S}$ such that $\mathcal{H}$ contains at least one element for each $\mathcal{S} \in \mathcal{C}$. A HS of $\mathcal{C}$ is minimal iff no proper subset of it is a hitting set of $\mathcal{C}$. In this work only the minimal hitting sets with one component are considered. They are called singles hitting sets.

2.1 Gröbner Bases

In order to obtain the CARCs of a system, we use Gröbner bases theory [3]. It 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 transform a set of equality polynomial constraints into a standard form. Having the set of equality polynomial constraints in the form $P = 0$, Gröbner bases produce an equivalent system $G = 0$ which has the same solutions as the original one but without some variables.

For example, if we have the constraints {`a+b-e=0, c+d-f=0, e*f-g=0`} and we want to obtain new constraints without the variables {`e, f`}, because these variables do not have sensors (they are non-observable variables). The obtained constraint is `(a+b)*(c+d)-g=0)`.

3 Storing a system in an ORCDB

Relational databases offer a way to store discrete information as *Integer*, *Date* or *String*. But in the case of diagnosis, where the components have different behaviours, it is necessary to look for another way to store and treat this type of information. Many database applications have to deal with infinite concepts like diagnosis. However, databases have a finitely capacity. The basic idea is that constraints can be used to represent, in a compact way, data that could be very large, or even infinite. CDBs extend the relational databases paradigm to allow working with infinite amounts of data with a finite representation.

There are other proposals for implementing and building prototypes for CDBs, whose main objective is handling spatial-temporal data. There are important proposals in this area as MLPQ/PReSTO [11], DEDALE [5] or CCUBE [2].

In general, neither of these prototypes offer a versatile solution for any type of application, only work in spatial-temporal environment, so they can not be used in model-based diagnosis. Another important disadvantage of these proposals is that they do not generate new polynomial constraints, and this is necessary for the diagnosis process.

In the work [4] an architecture that can be used in model-based diagnosis is proposed. This architecture is called ORCDB (Object Relational Constraint Database) because it stores equality polynomial constraints as objects in a relational database implemented with Oracle $9.\mathrm{i}^{TM}$. All the constraints are indexed by their variables to improve the computational time for obtaining the constraints related to some variables. This aspect of object-oriented databases indexation has been analysed in works as [1].

ORCDB architecture has three internal tables shown in Figure 1. These tables allow us to identify each polynomial equality constraint (table *Constraints*), each variable (table *Variables*) and to establish the relations between the constraints and the variables (*Constraint/Variables*), avoiding the study of all the constraints.

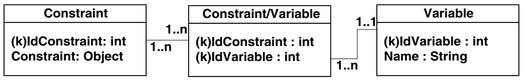

Figure 1. Tables to index constraints and variables

These three tables are in each ORCDB of the distributed system, and they are not available directly for the user. When a constraint data is added, indexes are created in the table *Constraint* in order to locate constraints more quickly and efficiently.

4 Distributed Architecture

The dare in this paper is to perform the same capacity of diagnose when the information is stored in distributed databases. In order to obtain this capacity we have developed an architecture where the SPM is distributed. Also it is important in our proposal to decide in execution time where the sensors are. This characteristic adds a new complication due to the components involved in the diagnosis process have to be determined.

As it is necessary to diagnose information stored in different databases, our architecture uses distributed databases, due to they have been developed to meet information needs of processes organizations engaged in distributed operations. It allows us that each node has its own set of components and requirements. Some of these involved data are used only in the node (private information), but other data are required as shared information (public information).

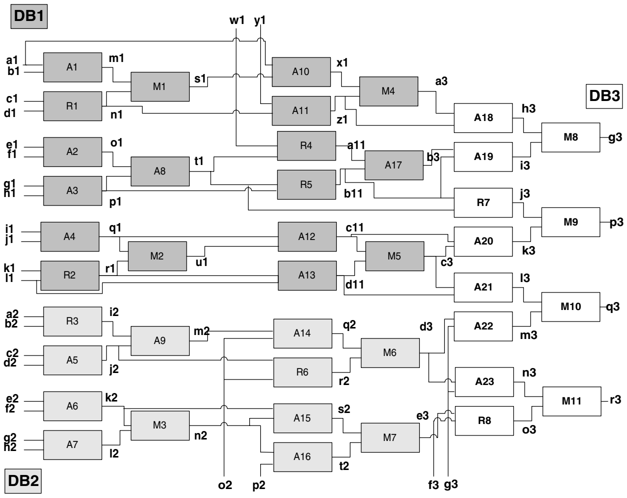
DB1
w1 y1
a1 b1 A1 m1 M1 s1 A10 x1 M4 a3
DB3
c1 d1 R1 n1 A11 z1 A18 h3 M8 g3
e1 f1 A2 o1 R4 a11 A19
A8 t1 A17 b3 i3
g1 h1 A3 R5 b11 R7 j3
p1 M9 p3
i1 j1 A4 q1 A12 c11 A20
M2 M5 k3
k1 l1 R2 r1 u1 A13 c3 d11 A21 l3 M10 q3
a2 b2 R3 i2 A9 m2 A14 q2 d3 A22 m3
c2 d2 A5 j2 M6 R6 r2 A23 n3
e2 f2 A6 k2 M3 A15 s2 e3 R8 M11 r3
g2 h2 A7 n2 M7 o3
l2 A16 t2
DB2
o2 p2 f3 g3

Figure 2. Distribute Diagnosis Example

In order to explain our proposal we are going to follow with the example shown in Figure 2, where there are three distributed databases, each component belong to an ORCDB depending on its colour. Where the components can be multipliers (M*i*), adders (A*i*) or subtractors (R*i*). Although these groups of components are presented joined in order to clarify their relation, there are three different systems physically separate (DB1, DB2 and DB3).

The distributed architecture is shown in Figure 3, where there are a central system and several distributed systems. The user asks to the central system about the diagnosis depending on the allocation of the sensors. The central system has a table where the relation between the ORCDBs (DB1, DB2 and DB3 for our example) and the variables is stored. The central system does not know what each database has, only their variables. Once the central system knows what databases have the information related with a query, the diagnosis is solved locally in each database. Some time it will be possible that each database can obtain CARCs in function of observable variables, but in other cases will be necessary the union among some CARCs of different databases. For this reason, the central system and each local system need a Gröbner bases solver, for local and distributed diagnosis.

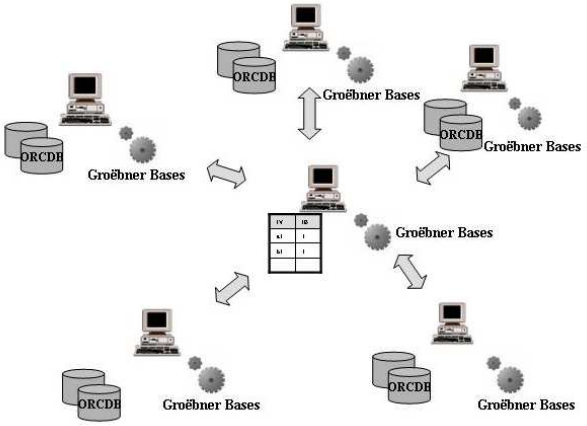

Figure 3. Architecture of the system

5 Distributed algorithm

The diagnosis process has four different phases to obtain the full diagnosis.

Each step of this algorithm will be explained in the next subsections with an example of sensors. For example if an engineer wants to know how are the CARCs of the distributed system of Figure 2, where the sensors are allocated in the variables $\{$a1, b1, n1, w1, y1, b3, p1, r1, q1, l1, g3, p3 and q2$\}$ to prove an OM.

5.1 Dividing the problem

This step answers the query about what databases are involved in the diagnosis for a group of sensors. The central system looks for the databases that have variables with sensors, using the table that relates databases and variables. For the example the three databases are involved.

In order to perform this division, it is necessary another definition. In distributed systems there are observable and non-observable variables, as in the normal model-base diagnosis, but there are other important variables (non-observable shared variables).

Definition 9. *Non-observable shared variables* are non-observable variables that are involved in the diagnosis and belong to two or more than two databases. It means that these variables can not be solved in the local system, but perhaps can be solved combining them with non-observable shared variables from other systems.

Thereby three different systems are created:

- **DB**1 with the observable variables {a1, b1, n1, w1, y1, b3, p1, r1, l1, q1} and {a3, z1, b11, t1, c3, c11, d11} as non-observable shared variables.
- **DB**2 with the observable variables {q2} and {d3, e3} as non-observable shared variables.
- **DB**3 with the observable variables {b3, g3, p3} and {a3, z1, b11, t1, c3, c11, d11} as non-observable shared variables.

5.2 Looking for the local CARCs:

In order to obtain the CARCs of each database, it is necessary works with the observable and non-observable shared variables. The idea is to find *Clusters of Related polynomial Constraints*, because it is not possible to create the Context Network with all the constraints of the database, due to not all the components have to participate in the SPM.

G≡ $\bigcup_i \{c_i\}$ is a cluster of related constraints (CRC) if $\forall$ c_i VarNoQueryOrShared(c_i)⊆VarNoQueryOrShared(G−c_i)

Where c_i is a polynomial constraint of the databases and VarNoQueryOrShared(*C*) are the variables of the constraints *C* that are non-observable or non-observable shared variables.

For the example the different CRCs are:

- DB1: {A1, M1, A10, A11 and A4}, {A8, R4, R5 and A17} and {M2, A12, A13 and M5}
- DB2 does not have CRC, because this database has only one observable variable.
- DB3: {A18, A19, M8, R7, A20 and M9}

Once the CRCs of each database are known, the CARCs can be obtained building the CN of CSs. But it is necessary a revision of the definition of CARCs because the non-observable shared variables can also participate in local CARCs.

Definition 10. *Context Analytical Redundancy Constraint (CARC)*: A constraint derived from the system, in such a way that only the observable or non-observable shared variables are related.

Now each database solves a local diagnosis problem, obtaining the CARCs and the PMCCs using the local CN:

DB1 with the CARCs:

−{a3 - a1n1 - a1n1^2 - a1y1 - b1n1y1=0} (A1, M1, A10, A11, A4) where a3 is a non-observable shared variable.

−{-a3 + a1z1 + a1n1z1 + b1n1z1=0} (A1, M1, A10, A4) where a3 and z1 are non-observable shared variables.

−{n1 + y1 - z1=0} (A11) where z1 is a non-observable shared variable. −{b3 + p1 - w1=0} (A8, R4, R5, A17) where all the variables are observable, so this is a local solution.

−{b11 - b3 - o1 - p1 + w1=0} (A8, R4, A17) where b11 is a non-observable shared variable.

−{o1 + p1 - t1=0} (A8) where t1 is a non-observable shared variable.

−{-c3 + l1 q1 + q1 r1 + l1 q1 r1 + q1 r1^2 } (M2, M5, A12, A13) where c3 is a non-observable shared variable.

−{-c11 + q1 + q1 r1=0} (M2, A12) where c11 is a non-observable shared variable.

−{l1+r1-d11=0} (A13) where c11 is a non-observable shared variable.

DB3 with the CARCs:

−{-a3 b11-a3 b3+g3-b11 z1-b3 z1=0} (A18, A19, M8) where a3, b11 and z1 are non-observable shared variables.

−{-b11 c11-b11 c3+p3+c11 t1+c3 t1=0 } (R7, A20, M9) where b11, c11, c3 and t1 are non-observable shared information.

As it is shown, each CRC forms a new bigger component combining the original ones. For example the CRC (A8, R4, R5, A17) forms a new components with several PMCCs as it is shown in Figure 4. This new component can have as variables the observable or the non-observable shared variables.

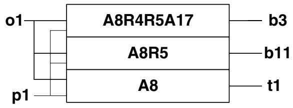

Figure 4. New Local Component (C2)

5.3 Combining solutions:

Once the parcial CARCs and their CSs are known, the central system builds a composition with all the new components and CARCs that depend of the the share variables in another database. The central system does not know the internal configuration of each database due to each CRC forms a new big component. The central system only knows

CARC1	CARC2	Single Minimal Hitting Sets
X	X	$C2\ (b11 \vee (b3 \wedge t1))$
X		$C1 \vee C4 \vee C2(b3)$
	X	$C3 \vee C5 \vee C2(t1)$

Table 1. Types of failures

the new bigger components as black boxes with input and output, the rest of the information is private, as it is shown in Figure 5.

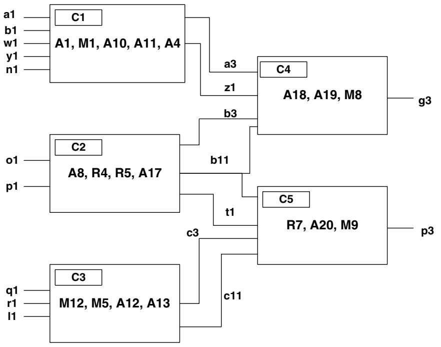

Figure 5. Cental System view of the example

In order to perform the model-based diagnosis the CN is created with all the possibilities. Although the only PMCCs are (C1C2C4) and (C2C3C5).

These new CARCs obtained only have observable variables, thereby it is possible to check the system with an OM. Depending on the OM three different situations are possible in function of the CARCs that are not satisfied, as it is shown in Table 1. The situations are that fails only CARC1 or only CARC2 or both at the same time.

5.4 Local Diagnosis

The local diagnosis is executed with the information received from the central system and with the local knowledge. Each local system has PMCCs associated to the CARCs. These CARCs can have non-observable shared variables, and in function of the message of the central system the local system can know what CARCs are failing and hence, the component that can fails. For example if the DB1 receives that t1 can be failing, this database can think that the component A8 are failing because the PMCC(A8) has the variable t1. It means that b3 and b11 have to fail too because they share the component A8. It is not possible because the local system has only received warning about t1, so the DB1 system send to the central system that it is not failing. It means that the mistake is in C3 or C5.

6 Conclusions and Future work

In this work we present a distributed architecture where constraints that represents equality polynomial behaviours are stored. These constraints are stored in several distributed ORCDBs where their variables are indexing to improve the computational time. In this paper it is also presented an algorithm in order to perform a distributed model-based diagnosis.

As future work we propose an extension of the algorithm to diagnose distributed systems avoid the use of central system, only with messages among the systems with ORCDB.

Acknowledgment

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] E. Bertino, B. Catania, and L. Chiesa. Definition and analysis of index organizations for object-oriented database systems. *Information Systems*, 23(2):65–108, 1998.

[2] A. Brodsky, V. E. Segal, J. Chen, and P. A. Exarkhopoulo. The ccube constraint object-oriented database system. In *SIGMOD '99: Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 577–579. ACM Press, 1999.

[3] B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. *Multidimensional Systems Theory, N. K. Bose, ed.*, pages 184–232, 1985.

[4] M. T. Gómez-López, R. M. Gasca, C. D. Valle, and F. F. de la Rosa T. Querying a polynomial constraint object-relational database in model-based diagnosis. In *DEXA*, volume 3588 of *LNCS*, pages 848–857. Springer, 2005.

[5] S. Grumbach, P. Rigaux, and L. Segoufin. The dedale system for complex spatial queries. In *SIGMOD Conference*, pages 213–224, 1998.

[6] J. D. Kleer. An assumption-based truth maintenance system. *Artificial Intelligence 28*, 2:127–161, 1986.

[7] J. D. Kleer, A. Mackworth, and R. Reiter. Characterizing diagnoses and systems. *Artificial Intelligence 56*, 2-3:197–222, 1992.

[8] G. Kuper, L. Libkin, and J. Paredaes. *Constraint Databases*. Springer, 1998.

[9] S. T. March and S. Rho. Allocating data and operations to nodes in distributed database design. *IEEE Transactions on Knowledge and Data Engineering*, 7(2):305–317, 1995.

[10] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence 32*, 1:57–96, 1987.

[11] P. Z. Revesz, R. Chen, P. Kanjamala, Y. Li, Y. Liu, and Y. Wang. The mlpq/gis constraint database system. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*, page 601. ACM, 2000.