CHEMICAL ENGINEERING TRANSACTIONS
VOL. 33, 2013
A publication of AIDIC
The Italian Association of Chemical Engineering
Online at: www.aidic.it/cet
Guest Editors: Enrico Zio, Piero Baraldi
Copyright © 2013, AIDIC Servizi S.r.l.,
ISBN 978-88-95608-24-2; **ISSN** 1974-9791
DOI: 10.3303/CET1333001

Research on Modeling Method for Constraint Inference

Qi Zhao, Kun Yang, Wenquan Feng*

School of Electronic and Information Engineering ,Beihang University, Xueyuan Road 37, Haidian District, Beijing, 100191, China
buaafwq@buaa.edu.cn

Fault diagnosis is of great importance especially in the field of aerospace, because spacecrafts are expensive and unique in most cases. Fault diagnosis can not only improve reliability of the spacecrafts but also reduce the workload of ground engineers, time of training astronauts and costs of lunching and running spacecrafts. Constraint inference is an important field in Artificial Intelligence and other areas of computer science. It has been widely used in model-based fault diagnosis, decision support, natural language understanding and other fields. Model-based fault diagnosis is an example of abductive reasoning using a model of the artifact. It has two steps: conflict identification and candidate generation. In the first step, the diagnosis system simulates the system using the model and compares the real observations with the predicted observations of the system. If conflicts are identified, we can get the reasons for the faults in the second step. Therefore, using constraint inference in model-based fault diagnosis, one significant step is building a model to simulate the real system. This article presents a new modeling method and uses this method to realize constraint inference in an example. It has been proved that this method is very convenient and efficient.

1. Introduction

Fault diagnosis is very important especially in the field of aerospace due to the expensive cost and uniqueness of spacecrafts. Using the results of fault diagnosis, ground engineers can supervise the running status of spacecrafts efficiently and timely (Giovani et al. 2009). Model-based method, as one of many fault diagnosis methods proposed in the last years, is widely used in aerospace field. Model-based fault diagnosis has two steps: conflict identification and candidate generation. In the first step, we use a model to simulate the real system and compare the real observations of the system with the predicted observations. If conflicts are identified, we can get the reasons for the faults in the second step. As an important field in Artificial Intelligence and other areas of computer science, constraint inference has a wide application in model-based fault diagnosis. Using constraint inference in model-based fault diagnosis, one significant step is to build a model to simulate the real system. This article first presents a new modeling method and then uses this method to realize constraint inference in an example.
We first use a text file to describe the real system. Different identifiers are used to distinguish various units, connection relationship between units, working status of each unit, input variables, and observed variables. Then a C++ program is used to analyze the text file and to get the information of the model. The information is then called by the constraint inference module, which could use constraint propagation technology to check whether the system has faults or not. Then the program could give the inference result. The advantage of this method is the convenience for users to change the model easily by modifying the model file directly without any changes of the program code. And the example proves that this method is efficient.

2. Methods

2.1 Model-based fault diagnosis

Model-based fault diagnosis is an example of abductive reasoning using a model of the artifact. It consists of two steps: conflict identification and candidate generation. As shown in Figure 1, by comparing the

Please cite this article as: Zhao Q., Yang K., Feng W., 2013, Research on modeling method for constraint inference, Chemical Engineering Transactions, 33, 1-6 DOI: 10.3303/CET1333001

predicted behavior, which is given by the inference result using system model, with the observed behavior, which is provided by the output of real system, we can know whether the system has faults. The diagnosis system will locate the faults as well.
This paper involves the first step, conflict identification, which is supposed to get minimum hitting set by constraint inference when expected behavior of a system differs from observed behavior.

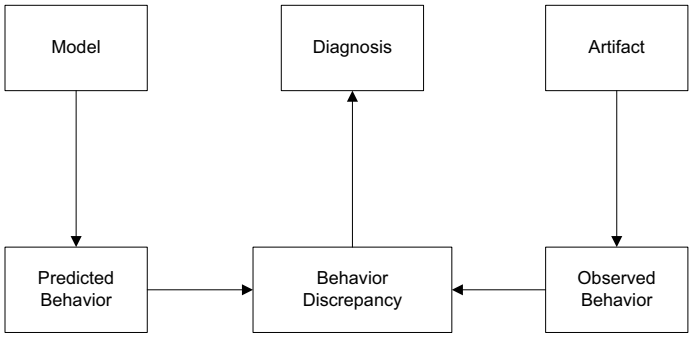


Figure 1: model-artifact difference

2.2 Constraint Satisfaction Problem
Constraint Satisfaction Problem, abbreviated as CSP, is extensively studied in AI (Kumar, 1992; Ji and Zhang, 2007). Its basic elements include V (Variable), D (Domain), and C (Constraint). Therefore CSP can be described as a VDC problem. D is a set including all possible values of variables; C describes the relation between variables. Based on D's difference, CSP could be divided into three categories, Boolean CSP, finite CSP and mixed CSP.
Variables in Boolean CSP only can be 0 or 1, which means D in Boolean CSP is the set {0, 1}. In fact C in Boolean CSP is a group of propositional logic formulas. Variables is 0 or 1, therefore the value of formula that is consisted of Boolean variables is 0 or 1. To solve Boolean CSP we should assign all variables to make the value of each constraint is 1.
Domain in finite CSP is finite, for example value of variables can be integer. In this case we usually ignore concrete form of constraint and just list all possible values satisfying the constraint, like N-Queen problem (Russell and Norvig, 1995).
Domain in mixed CSP can be Boolean, finite or infinite. Like Boolean constraint, value of mixed constraint is true or false. To solve this problem, we give a group of assignments for variables to satisfy all constraints. We use mixed constraint in this paper.

2.3 Representation of CSP
Propositional logic makes it possible to represent CSP. Using propositional logic we can transfer a problem described in natural language into a problem described in formal logic language. Then inference can be conducted based on logic calculus laws. As mentioned above, C in Boolean CSP is a group of propositional logic formulas. Actually constraint of CSP can be described by propositional logic formulas.
There is a theorem in the area of propositional logic, which is any propositional formula can be converted into conjunctive normal form (CNF) or disjunctive normal form (DNF). The form of CNF likes $C1 \wedge C2 \wedge \cdots \wedge Cm$. Ci (i=1,2,…,m) is called clause and its form likes $L1 \vee L2 \vee \cdots \vee Ln$. Lj (j=1,2,…,n) in above formula is called literal. Similarly, the form of DNF is $C1 \vee C2 \vee \cdots \vee Cm$. Ci (i=1,2,…,m) is clause and its form likes $L1 \wedge L2 \wedge \cdots \wedge Ln$. Lj (j=1,2,…,n) in above formula is literal. Therefore, an inference problem can be described by CNF or DNF. We use CNF to present CSP in this paper.

2.4 Constraint inference
Constraint inference is an important concept in the field of constraint program (Liu, 2008). It is widely used and includes all kinds of methods of solving CSP. Constraint Inference have two aspects, constraint satisfaction searching and constraint language. Constraint satisfaction searching includes backtracking, constraint propagation, intelligent backtracking and true value maintenance, local correction method and so on. Constraint language includes CONSTRAINTS、CHIP、COPS and ILOG (Zhou et al., 2008).
Constraint propagation is a key constraint inference method. Researchers come up with a lot of constraint propagation algorithms since it was presented. At present some algorithms are widely used, such as consistence checking, tree of decomposition, bucket elimination, unit propagation and so on.

The idea of constraint propagation is reasoning unknown variables based on known variables and at the same time the dependency relationship between variables are recorded. We use CNF to represent the problem, and then conduct constraint propagation. This means that each literal will be traversed. During the traversal, if the value of one variable is the only unknown one in the literal it is included, as well as the value of this literal is the only unknown one in the clause it is included, the unknown variable will be assigned. Then this variable can be known one and constraint propagation continues.
Constraint propagation stops in one of the below situations:
(1) Conflict. It means that the inference value differs from observed value.
(2) All variables are assigned.
(3) Not all variables are assigned because the incompletion of constraint logic.

3. Result

3.1 Framework

The framework has three modules, which are: model analyzing module, model building module and constraint propagation module, as shown in Figure 2.

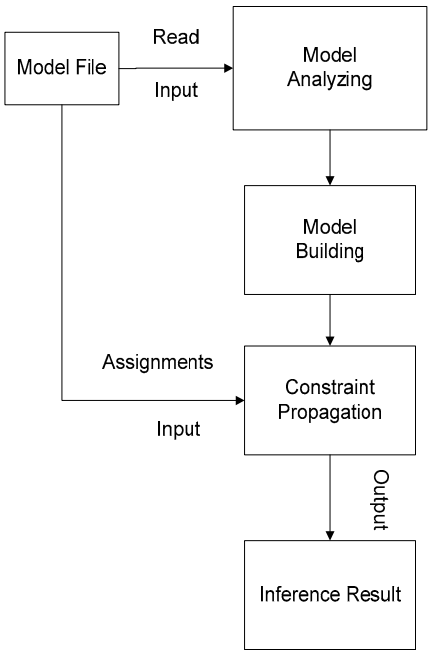


Figure 2: framework of simulation

Model file (.txt) contains all the information of the system. Model analyzing module reads model file and analyzes the information. We could change the model easily by modifying the model file directly and get model information. Model building module reads model information and builds the inference model. A group of assignments of variables will be given by model file. Based on these values, constraint propagation module conducts inference to get the result.

3.2 Model file

Model file describes the structure of the system and contains all the information of the system. Model file is read by model analyzing module, and then information will be stored.
Different identifiers are used in the model file to distinguish various units, connection relationship between units, working status of each unit, input variables and observed variables. These identifiers are explained as follows:
(1) "#" is an identifier, which identifies the name of model;
(2) "/" is an identifier. Strings after "/" will be recognized as comment, which will not have any influence on model information.
(3) "@" identifies the unit in the model. The numbers after "@" mean unit ID. The number after unit ID reflects the working status of the unit. The last part presents the type of the unit, which can be basic

logic unit or combinational logic unit. For example, “@1, normal input” means unit 1 is an input of the system and works in normal status. In this program, each unit has three working status, normal, stuck_1 and stuck_2. While the unit works in normal status, it gives the correct result. When it works in stuck_1 status, its output will be always 0. When it works in stuck_2 status, its output will be always 1.

(4) “%” identifies the connection between units. Two numbers, connected by “_”, are the ID of each unit. The output of the former unit connects the input of the later unit. For example, “%1-5” means the output of unit_1 connects with the input of unit_5.

(5) “$” means an assignment of input of the unit. For example, “$1, 1” means the value of input of unit_1 is 1.

(6) “&” means the observation port and its value. For example, “&6, 4” means we observe unit_6 and its output is 4.

Therefore, it is very convenient to use this model file configurating model information. The only thing we need to do is to modify this file rather change program code.

3.3 Realization of constraint inference

We use C++ programming to realize constraint inference. A function named ConInf() is used to conduct constraint propagation, and it will return a COMP value as the result of inference.

Firstly, this function will check the number of unknown variable of each unit. If the number is 1, the input ports and output ports of the unit will be checked to confirm which port is unknown. After that procedure the unknown port will be assigned a value. In a word, each literal of each clause will be checked and the only unknown variable will be assigned for a value.

However, there is a possible situation when we get the output value of a unit under its predicted working status, but we may get the same result under its other working status. For example, if the working status of a unit is stuck_2, its output will always be 1 no matter what the input is. We assign an input value for this unit, and we can get its output value is 1 if we predict its working status is normal. In this case we cannot confirm which status is accurate. So we need to give another assignment to conduct constraint inference again.

The order of constraint propagation is determined by the order of building unit in the model file. Once a unit is built, the ID of this unit will be inserted into unitMAP. During constraint propagation each unit is traversed by order of unit ID. Constraint propagation will stop if the predicted status is incorrect or the program gets an inference result.

3.4 An example

The model of the example is shown in Figure 3. There are five input ports, four adders, two multipliers, one comparator and one OR gate in this model.

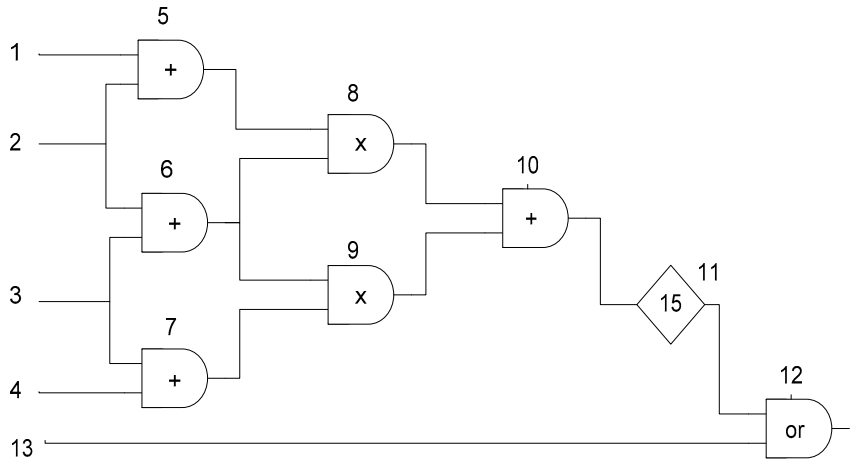


Figure 3: model in the example

We use a model file to describe the structure of above model. Then the C++ program will load the model file, analyze the information and conduct constraint inference. We will get different results if we configurate model file with different parameters. There are three possible situations:

(1) consistency satisfied

For example, we assign unit_1's value is 1, unit_2's value is 2, unit_3's value is1, unit_4's value is 4 and unit_13's value is 0. We choose unit_12 as the observed port and its output value is 1. Then we assign unit_5 works in stuck_2 status and other units work normally. Running the C++ program we can get the inference result, as shown in Figure 4A. The result "equal!" means the predicted value is the same as the observed value.

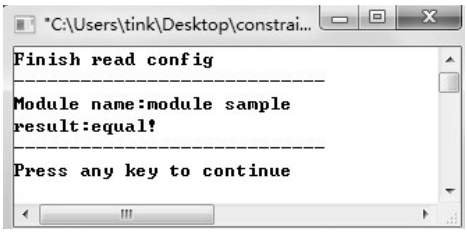
(2) confliction

It is clear that if the observed parameter is assigned for other value, except 1, there will be fault, because the predicted value is different from the observation. Simply, we change the observed value of unit_12 to 0 for example, and then we run the C++ program to get inference result. Inference result is shown in Figure 4B. The program also tells us which unit there appears a confliction.

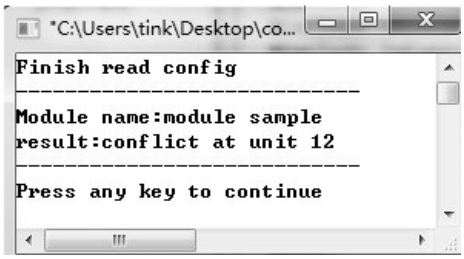
(3) Inference confusion

Just as motioned above, there is a situation when we cannot get an inference result directly. We can assign the input value of unit_2 as 0 to simulate this situation. In this case, unit_5's state is stuck_2, so its output value is always 1. We can also know that the inputs of unit_5 are 0 and 1. If its state is normal, the output value is also 1. Therefore, we cannot judge the accurate state of unit_5. Another word, we cannot know the state of unit_5 is stuck_2 or normal. User will be asked to input another assignment to conduct inference.

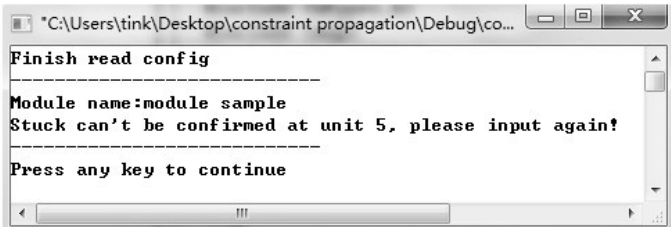
From Figure 4C we can see that the program tells us "Stuck can't be confirmed at unit 5, please input again".



A: consistency satisfied



B: conflict



C: inference confusion

Figure 4: result of inference

4. Conclusion

This paper introduces diagnosis-related methods about constraint inference, including CSP, propositional logic, CNF, constraint propagation and so on. A novel and simple modeling method is represented, which is very convenient and efficient to modify model by configurating model file. Also, an example is given to illustrate this method and it gives correct result.

The advantage of this method is that users do not need to change the program and they just need to configurate the model file directly. Meanwhile, it is very easy to build more complex model.

References

Giovani S. C. da Silva, Mauricio B. de Souza Jr, Enrique L. L. Mario C. M. M. C. 2009, Application of a Model-based Fault Detection and Diagnosis System to a Hydrotreating Reactor, Chemical Engineering Transactions, 17, 1329, DOI: 10.3303/CET0917222.

Ji X.H., Zhang J., 2007, Constraint Processing, acat automatic sinica, 33(2), 125-131.

Kumar V., 1992, Algorithms for constraint-satisfaction problems: A survey, AI Magazine, 13(1), 32-44.

Liu C.H., 2008, Research on the Constraint Propagation Based Constraint Solving Methods, Jilin University, Changchun, China

Russell S.J., Norvig P., 1995, Artificial intelligence: a modern approach, Prentice Hall, London, United Kindom.