International Journal of Control and Automation
Vol.7, No.6 (2014), pp.269-284
http://dx.doi.org/10.14257/ijca.2014.7.6.26

# A Conflict Diagnosis Approach of Changing Sequences in Gene Ontology Evolution

Qiu Ling[1], Liu Yong[2], Song Yingjie[3,4] and Zhang Bin[3]

[1] *School of Computer Science, Sichuan University of Science and Engineering, 643000 Zigong, China*
[2] *Sichuan University of Science and Engineering School of Automation and Electronic Information, 643000 Zigong, China*
[3] *School of Computer Science and Technology, Shandong Institute of Business and Technology, 264005 Yantai, China*
[4] *Artificial Intelligence Key Laboratory of Sichuan Province, Sichuan University of Science and Engineering, 643000 Zigong, China*

*tiantianyingjie@gmail.com (Corresponding author)*

***Abstract***

*The goal of the Gene Ontology is to provide a controlled vocabulary that can be applied to all organisms, even as knowledge of gene and protein roles in cells is accumulating and changing. Further, in the process of co-evolution, conflicts are unavoidable among ontology changing sequences. We use the $N \times M$ matrix to model the changing sequences in Gene ontology evolution which lead to a result ontology. According the users' estimate of the changing sequences, we propose the updating algorithm to update the matrix as mentioned earlier. The new matrix is as the input of the MUOUE((A,R) algorithm to calculate the minimal hitting sequence set of high-potential candidates that were subject to a subsequent Bayesian reasoning approach to rank them. The instances show that our approach can be used to diagnosis the conflicts, especially under the opening environment. The diagnosis approach we proposed in this paper can provide support for the co-evolution of Gene Ontology.*

***Keywords:*** *Gene Ontology Evolution; Conflict Diagnosis; Bayesian reasoning*

## 1. Introduction

The goal of the Gene Ontology (Go) [1-4] is to provide a controlled vocabulary that can be applied to all organisms, even as knowledge of gene and protein roles in cells is accumulating and changing. The project was a collaboration between three model organism databases: FlyBase (Drosophila) [5], the Saccharomyces Genome Database (SGD) [6] and the Mouse Genome Database (MGD) [7]. There is one central place were the ontology resides, but the control is distributed. There are a few full time "curators" that work on the vocabulary and the relations. Go users can make suggestions for additional terms or for other improvements via change requests. Eventually, the curators perform the changes. Changes occur on a daily basis. Over time, GO has grown larger, has become more complex, and has grown in breadth. A record is kept of every change, and there are procedures that check for local conflicts in the ontology

ISSN: 2005-4297 IJCA

evolution. Initially, the changes were managed via CVS, later on a database was used. The database allows full roll-backs of changes.

CVS[8-9] and database use locking mechanism to avoid conflicts, but in fact, the conflicts can not be avoided, because ontology will be simultaneously edited by more than one person in the opening environment. It is a social process [10]. There are many participators and most of them are not aware of the existences of each other at all, let alone communicate enough, this exacerbates the problem [11-12]. The rich semantic relationships in ontology model exacerbate the situation, since different pars of the ontology model have much more complicated impacts on each other. In addition to the increasing Gene Ontology [13-14] complexity, manual debugging the root of conflicts is extremely expensive in terms of labor cost, diagnosis the conflicts among changing sequences by engineers is impossible. These conflicts may cause the changing ontology to critically fail or life-threatening consequences. So it is necessary to research the approaches for the changing sequences conflicts diagnosis. Automatic software fault location techniques [15-18] aid developer to pinpoint the root cause of conflicts, thereby reducing the debugging effort.

In the previous works [19-20], we have arrived at a solution via scientific investigation for ontologes consistency reasoning. We propose a new approach to interpret ontologies document in a lightweight modeling language for software design, which is used to provide a non-standard reasoning service for the verification of ontologies. Motivated by the challenges of Gene Ontology evolution and based on the previous works, in this paper, we present a methodology for ontology evolution, by focusing on the conflicts diagnosis of multi-participant. The instances show that our approach can be used to diagnosis the conflicts, especially under the opening environment.

In this paper, we focus on the conflicts diagnosis of the changing sequences in the Gene Ontology evolution. Our main contributions are summarized as follow.

1. We express the changing sequences of Gene Ontology Evolution in manner of a matrix. Then, we update the matrix according to users' evaluation.

2. We propose a new algorithm to compute the Minimal hitting sequence set of the changing sequence matrix.

3. Bayesian reasoning approach is used to rank the candidate diagnosis of the Minimal hitting sequence set.

## 2. Concepts and Definitions

**Definition** 1 **(Ontology Changing).** The Ontology Changing can be defined as

$$OC := (O, CS, ECS, COM, O')$$

where:

- $O$ is the original ontology.
- $CS$ is a set of different changing sequences which come from different users.
- $ECS$ represents expanded changing operations $ECS = CS \cup AC$. To convey changing semantic sufficiently and reduce the effect of ontology evolution, there are always some additional changing operations, which is denoted by $AC$.

 

– *COM* is the consistent ontology model. If an ontology *O* is not satisfied with *COM (O⊭ COM)*, it is inconsistent. Else it is consistent.

– *O'* is the result ontology.

**Definition 2 (Ontology Changing Sequence)**. The Ontology changing sequence can be defined as

$$Chs=< Och_1, Och_2, \ldots Och_n>,$$

which is composed of a succession of Ontology Changes $Och_1$, $Och_2$, $\ldots Och_n$. The adjacent changes are separated by ",". If there are two changes $Och_i$ and $Och_{i+j}(j>=1)$ , $Och_i$ is the predecessor of $Och_{i+j}$, and the order of the change sequence can't be changed.

**Definition 3** (**Ontology Changing Operation**). The Ontology Changing Operation *Och* is defined as a 4-tuple:

$$Och_{user}:=\{id, operation, subject, constraint\},$$

where

– *id* is used to as a unique identification of the changing sequence's sequencing.

– *operation* is the basic type of *Och*, which are {*Add, Delete, Modify*}.

– *subject* $\subseteq \mathcal{O}. C \cup \mathcal{O}. P$, means that the subjects of operation are {*Concept, Property*}.

– *constraint*$\subseteq \mathcal{O}. C \cup \mathcal{O}. P$, is used to indicate contextual constraints.

**Table 1. The Atomic Change of Ontology**

| *Operation* | description | extension |
|---|---|---|
| *AddConcept*($C_1$, $C_2$) | Add concept $C_1$ as the subconcetp of $C_2$ | Not delete $C_2$ |
| *DeleteConcept*($C_1$, $C_2$) | Delete concept $C_1$ whose supconcetp is $C_2$ | $C_1$ is subconcept of $C_2$ |
| *ModifyConcept*($C_1$) | Modify concept $C_1$, contain renaming and the modification of its attributes | |
| *AddProperty*($P_1$, {$C_1$,$C_2$}) | Add property $P_1$, whose *domain* is $C_1$ and *range* is $C_2$ | Not delete $C_2$ and $C_2$ |
| *DeleteProperty*($P_1$) | Delete property $P_1$ | |
| *ModifyProperty*($P_1$, {$C_1$,$C_2$}) | Modify property $P_1$, whose new *domain* is $C_1$ and new *range* is $C_2$ | Not delete $C_2$ and $C_2$ |

Table 1 shows ontology change operations which involved in this paper. These operations refer to the atomic ones. In the table 1, there are three types atomic ontology changes, which are *Add*, *Delete* and *Modify*. Hürsch [21] divide atomic changes into *Add* and *Delete*. We add *Modify* as the third one. For that, the operation *Modify* can be broken down as delete the modified concept first and then add the objective concept outwardly. However, in the actual evolution the process is inadvisable. For example, under the assumption of *Modify* is nonexistent, if we want to rename the concept $C_1$ as $C_1'$. Firstly, the concept $C_1$ should be deleted, if there is a subconcept $C_2$ of $C_1$, it will

also be handled, which may be deleted or as subconcepts of $C_1$ 's supconcept $C_0$ . Then $C_1'$ will be added to replace $C_1$. Obviously, the hiberarchy of the ontology has been destroyed and the process doesn't satisfy the requirement.

The last column of the table is extensions of changing operations, it can also be understood as the constraints when the changes take place. It reveals the implicit semantics. For example, the add operation *AddConcept*($C_1$, $C_2$) in the first row has extension that the concept $C_2$ can't be deleted. We can see from the description of the operation that the semantic of the operation is to add the concept $C_1$ as a subconcept of $C_2$. If there is an operation of deleting the concept $C_2$, and the operation of delete concept $C_2$ take place earlier than the operation of adding operation *AddConcept*($C_1$, $C_2$), then the added concept $C_1$ will become an isolated concept in the ontology under the condition with no extensions, which is unreasonable. The other extensions are the similar. However, not all operations need extension, e.g. the operation of modifying the concept *ModifyConcept*($C_1$) in third row and the operation of deleting property *DeleteProperty*($P_1$) in the fifth row.

**Definition 4** (**Conflict Changing Sequence**).

If a changing sequence $Chs$=< $Och_1$, $Och_2$, …$Och_n$> acts on the original ontology $O$, and the result ontology $O' \nvDash COM$, we say that $Chs$ is a Conflict Changing Sequence (*Ccs* for short). And there may be one or more changing operations in the *Ccs*, which express as {$Och_i$, $Och_j$,…$Och_k$ } (1<=i, j,…k<=n), are the root of the ontology inconsistency.

## 3. Conflicts Diagnosis Model

In this section, we first present a conflicts diagnosis model and a matrix updating algorithm. Then, we propose an algorithm to derive an approximate collection of the minimal hitting sequence set of the updated matrix. However, the minimal hitting sequence set is not the final diagnosis. We used the Bayesian reasoning approach to rank the elements of it in order of likelihood to be the diagnosis.

We use the following function to represent the diagnosis model.

$$O' = \mathcal{F}(Chs, r)$$

Where $Chs$ and $O'$ represent input and output of an Gene ontology evolution, respectively, and where $r = (r_1, r_2, \ldots r_m)$ indicates the state of the result ontology. For each changing operation $Och_i$, the binary states are: $r_i = 1$ ($O' \vDash COM$) or $r_i = 0$ ($O' \nvDash COM$). Conflicts diagnosis can be understood as solving the inverse problem $r = \mathcal{F}^{-1}(Chs, O')$, *i.e.*, finding the combinations of states of result ontology that explain the observed output for a given input.

In the remainder of the paper, the set of the Changing Sequences in the Gene Ontology evolution is encoded into a $N \times M$ matrix $A$. Where $N$ is the number of the changing sequences, and $M$ is the number of all involved changing operations. The element $a_{ij}$ is equal to 1 if operation $Och_j$ ($j <= M$) is a member of the changing sequence $i$ ($i <= N$), and 0 otherwise. Let $R$ denote the result ontology vector, where $r_i$ signifiers whether the changing sequence $A_{i*}$ has a consistent result.

As is shown in matrix (M1), $N = 2$, $M = 3$, and the result ontology vector contains two elements. Each row corresponds to a changing sequence, exemplified by the first row in which operations $Och_1$, $Och_3$ are involved, and the corresponding result $r = 1$, which mean that the result ontology is inconsistent. The matrix ($A$, $R$) can be partitioned

 

into two sets with respect to the result verctor. One set consists of consistent result ontologies $A_{con} = \{ A_{i*} \mid r_i = 1\}$, and the other contains the inconsistent result ontologies $A_{inc} = \{ A_{i*} \mid r_i = 0\}$. The set $A_{inc}$ is used to derive the set of diagnostic candidates.

$$\begin{array}{c} \quad Och_1 \quad Och_2 \quad Och_3 \quad R \\ \begin{array}{c} A_{1*} \\ A_{2*} \end{array} \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{array} \tag{M1}$$

In opening environment, the evolution requirements of Gene Ontology keep adding. To represent developers' different preferences, we propose an algorithm based developers' evaluation to update the matrix. In an optimal situation, a developer *u.eval* only evaluates one changing sequence with the corresponding row in the matrix $A_{i*}$.In such a case, we will update all elements, whose value is not equal to 0, with their value plus 1. Otherwise, it indicates that the the user's preference is not contained in the matrix, and the changing requirement will be added into the matrix as a new row. We get the updated matrix using algorithm 1 MUOUE((*A*,*R*), *u.eval*).

**Algorithm 1** The **M**atrix **U**pdating based **O**n the **U**ser's **E**valuation((*A*,*R*), *u.eval*) (MUOUE((*A*,*R*), *u.eval*) )

Input: the changing sequences matrix (*A*,*R*), and a developer's evaluation *u.eval*
Output: the updated matrix (*A'*,*R'*)

1. $A' \leftarrow A$;
2. $seq \leftarrow \phi$;
3. flag = 0;
4. **for** all i∈{1…*N*}
   a) **if** $A_{i*}$ is the evaluated changing sequence **then**
   b) $seq \leftarrow A_{i*}$;
   c) flag = 1;
   d) **Otherwise** $seq \leftarrow u.eval$;
   **e)** **endif**
5. **endfor**
6. **if** flag = 1 **then**
   a) **for** j∈{1…*M*}
      i. **if** $seq_j \neq 0$ **then**
      ii. $seq_j$+=1;
      **iii.** **endif**
   **b)** **endfor**
   c) $A'_{i*} \leftarrow seq$;
7. **else**
   a) $A' \leftarrow A' + seq$ ;
   b) **if** $O_{A'N+1*} \vDash COM$ **then**
      i. $r_{N+1} = 0$;
   c) **else**
      i. $r_{N+1} = 1$;
   **d)** **endif**
8. **endif**
9. Return (*A'*,*R'*);

Exemplified by the previously mentioned matrix, there is a evaluation of $A_{2*}$. According to the algorithm 1, the matrix will be updated as matrix (2) shown.

$$\begin{array}{c} \\ A_{1*} \\ A_{2*} \end{array} \begin{array}{c} \begin{array}{ccc} Och_1 & Och_2 & Och_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 2 \end{pmatrix} \end{array} \begin{array}{c} R \\ \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{array} \quad \text{(M2)}$$

**Definition 5 (Minimal Hitting Sequence)**

Let *Ccs-set* is a non empty set, which contains *n* conflict changing sequences *Ccs-set*= {$Ccs_1$, $Ccs_2$,…$Ccs_n$} (*Ccs-set* = $A_{inc}$). For each $Ccs_i \in$ *Ccs-set* ($i$ = 1, 2,…, n), it includes not more than *M* ordered changing operations $Och_j$ (j∈{1,2,…M}). We define the Minimal hitting sequence *d-Chs* of the *Ccs-set* as:

$$\forall\, Ccs_i \in \text{Ccs-set},\ Ccs_i \cap \text{d-Chs} \neq \phi \wedge \nexists \text{d-Chs'} \subset \text{d-Chs}: Ccs_i \cap \text{d-Chs'} \neq \phi$$

It can be interpreted as, given a set of conflict changing sequences *Ccs-set*, whose changing operations from universe $U = \cup Ccs_i = \{Och_1, Och_2, \ldots Och_m\}$, a Minimal hitting sequence *d-Chs* is a set $S \subseteq U$ such that $S \cap Ccs_i \neq \emptyset$, for all $i$ ($i$ = 1, 2,…, n), i.e., a set which contains, at least, one changing operations.

**Definition 6 (Minimal Hitting Sequence Set)**

Generally, there are more than one Minimal hitting sequences, we use *D-Chs* to represent the Minimal hitting sequences set *D-Chs* = {$d\text{-}Chs_1$, $d\text{-}Chs_2$,…$d\text{-}Chs_{|D\text{-}Chs|}$}.

According to the above descriptions and definitions, we come to the following corollaries.

**Corollary 1**. If there is a operation presents in all rows, whose result vector $r_i$ = 1, then the operation is a minimal hitting sequence.

**Proof** (The sufficiency). For the operation appears in all rows whose result vector $r_i$ = 1, so it has intersection with all conflict changing sequences, and it dosen't subsume nonvoid proper subset, so it is the minimal.

**Corollary 2**. If there is an operation does not occur in any row, whose result vector $r_i$ = 1, then it is impossible as an element of any minimal hitting sequence.

**Proof** (The sufficiency). Because the operation does not occur in all rows, whose result vector $r_i$ = 1, that is to say, the operation dosen't have intersection with any conflict changing sequence, if it appear in the minimal hitting sequences sets, this assumption violates the definition of the minimal hitting sequence. Then it is impossible as an element of any minimal hitting sequence.

**Corollary 3**. If there is a row of the matrix $A_{i*}$, all operations of $A_{i*}$ equal to 0, and $r_i$ = 1, then there are no minimal hitting sequence solution. Otherwise, for any row $r_i$ = 1, there must be some elements equal to 1, then there must be a minimal hitting set solution.

The first case: If there is a row of the matrix $A_{i*}$, all operations of $A_{i*}$ equal to 0, and $r_i$ = 1, then there are no minimal hitting sequence solution.

 

**Proof**: Suppose that there is a solution of the matrix. If the solution is {*d-Chs*}, then according to definition 5 ($\forall$ $Ccs_i \in Ccs\text{-}set$, $Ccs_i \cap d\text{-}Chs \neq \phi$), the conjunction of *d-Chs* and any row ($r = 1$) is not null. However, as mentioned above, for all $j \in M$, $A_{ij} \in A_{i*}$, $A_{ij} = 0$, and the conjunction of any set and empty set is null, which is discrepant with the assumption.

The second case: for all rows $r_i = 1$, which must have some operations equal to 1, then there must be minimal hitting set solution.

**Proof**: Assume that there is no solution for the matrix. Then $\forall i \in N$, $Ccs_i \in Ccs\text{-}set$, $Ccs_i \cap d\text{-}Chs \neq \phi$ is false. However, if there is a set of operations *d-Chs*, which contains all changing operations of the matrix ($\forall j \in M$, $Och_j \in d\text{-}Chs$), and because for all rows $r_i = 1$, there must be some operations equal to 1, which means that for any row $r_i = 1$, the conjunction of *d-Chs* and the row is not null, then we believe that *d-Chs* can be seen as a solution of the matrix. That is contradictory.

A brute-force approach to compute the minimal hitting sequence set *D-Chs* for the matrix would be to iterate through all possible component combinations to ① check whether it is a hitting set, and ② (if it is a hitting set) whether it is minimal, *i.e.*, not subsumed by any other set of lower carinality (cardinality of a hitting set *d-Chs*, |*d-Chs*|, is the number of operations in the set). As all possible combinations are checked, the complexity of such an approach is $O(2^M)$. So the brute-force algorithms have a cost that is exponential in the number of operations.

As mentioned in the previous section, we propose a matrix-decomposition based approach, which named(CMHSS(*A*,*R*)), to calculate the minimal hitting sequence set. The key idea behind our approach is the fact, that operations that are involved by more conflict changing sequences than others, may be an indication that there is a minimal hitting sequence set containing such operation.

**Algorithm 2** The **C**alculation of the **M**inimal **H**itting **S**equences **S**et of (*A*,*R*) (CMHSS(*A*,*R*))

Input: the $N \times M$ matrix (*A*,*R*) which have been updated based on the users evaluation
Output: the minimal hitting sequence set *D-Chs*

1. *D-Chs* ← ∅; temp = 1; Temp(*A*,*R*) ← ∅;
2. **For** $i \in \{1 \ldots N\}$ **do** {
   a) If ($r_i = 0$)
   b) (*A*,*R*) ← (*A*,*R*) - $A_{i*}$;
   c) $N{-}{-}$;
   }
   **Endfor**
3. **For** $i \in \{1 \ldots N\}$ **do** {
   a) **If** ($\sum_{j=1}^{M} A_{ij} = 0 \wedge r_i = 1$) **Then**
      i. Return Fault;
      **Endif**
   }
   **Endfor**
4. **For** $j \in \{1 \ldots M\}$ **do** {
   a) **If** ($\sum_{i=1}^{N} A_{ij} = 0$) **Then**
   b) (*A*,*R*) ← (*A*,*R*) - $A_{*j}$;
   **Endif**
   }
   **Endfor**
5. **For** $j \in \{1 \ldots M\}$ **do** {

a) **For** (temp=1,i =1; i<= N, i++)$i \in \{1 \ldots N\}$ **do** {
b) temp = temp * $A_{ij}$ ;
}
**Endfor**
c) **If** (temp ≠ 0) **Then**
i. *D-Chs* ←$Och_j$;
ii. $(A,R) \leftarrow (A,R) - A_{*j}$;
**Endif**
}
**Endfor**
6. **If** the matrix $(A,R) \neq \emptyset$ **then**
a) $m_j$ = Max ($\sum_{i=1}^{N} A_{ij}$| ($j \in \{1 \ldots M\}$));
7. **Else** goto line 11;
**Endif**
8. Temp$(A,R) \leftarrow (A,R)$;
9. **For** $i \in \{1 \ldots N\}$ **do** {
a) **If** ($A_{ij}$ = 1) **Then**
i. Temp $(A,R) \leftarrow$Temp$(A,R) - A_{i*}$;
**Endif**
}
**Endfor**
10. *D-Chs*' = MHSS (Temp$(A,R)$);
11. Return *D-Chs*;
}

---

There are 4 loop functions in the algorithm's primary stage. The first loop (line 2) is used to delete the changing sequences without conflict, then the result matrix contains only conflict changing sequences. The second one (line 3) is to check whether there is a minimal hitting sequence set solution, it is used for the reduction of time of calculating non-solution matrixes. The next one (line 4) is the traversal of the columns of the matrix. If there is a column whose all elements are 0, according to corollary 2, it is impossible as an element of any minimal hitting sequence. So we delete the column from the matrix. The last loop (line 5) is also the traversal of the columns of the matrix, if there is a column whose each element is 1, the corresponding operation is a minimal hitting sequence of candinality 1 as as described in corollary 1.

From the 6th line, we decompose the matrix. The row selection of matrix decomposition is to exploit a ranking based on the number of sequence involvements such as (line 6.a)).

$$A\left(Och_j\right) = \sum_{i=1}^{N} A_{ij} \tag{F1}$$

The algorithm is recursive, for CMHSS$(A,R)$ algorithm is called again in line 10 with this decomposed matrix as the input. The solution will combinewith the operation which is the basis of the matrix decomposition. Our algorithm avoids to iterate through all possible component combinations (O($2^M$)).

## 4. Ranking Diagnoses

While the above algorithm increases search efficiency, the number of minimal hitting sequences can be prohibitive, while often only a subset need be considered that are

 

most relevant with respect to the conflicts.The result *D-Chs* of algorithm 2 only be look upon as diagnosis candidates that are consistent with the observation matrix. But just cardinality will not suffice. Though there are many minimal hitting sequence, but not all of them are equally probable. Hence, the computation of diagnosis candidate probabilities Pr(*d-Chs*), and ranking them, is critical to the diagnostic of our method.

In our method, diagnosis candidates *d-Chs* are ranked in order of probability Pr(*d-Chs*) of being true conflict explanation. It is computed using the Bayesian probability update, which is used as the foundation of computing the posterior probability Pr(*d-Chs*) of the candidate *d-Chs* being the actual conflict diagnosis. For the probability of each candidate describe the actual evolution state, so we extend the computation to the updated matix, which contains both conflict and non-conflict changing sequences. The Bayes' update according to formula (F2).

$$\Pr(d-Chs_k \mid A_{i*}) = \frac{\Pr(A_{i*} \mid d-Chs_k)}{\Pr(A_{i*})} \cdot \Pr\left(d-Chs_k \mid A_{i*-1}\right) \tag{F2}$$

Where $A_{i*}$ is the $i^{th}$ row of the matrix. The denominator Pr($A_{i*}$) is a normalizing term that is identical for all *d-Chs* and thus needs not be computed directly. Pr($d$-$Chs_k$|$A_{i*-1}$) is the prior probability of Pr ($d$-$Chs_k$|$A_{i*}$). The specific calculating process will be covered later in this paper. The numerator Pr($A_{i*}$| $d$-$Chs_k$) is defined as formula (F3)

$$\Pr\left(A_{i*} \mid d-Chs_k\right) = \begin{cases} 0 & \text{if}(A_{i*} \wedge d-Chs_k \text{ ' } \perp) \\ 1 & \text{if}(d-Chs_k \rightarrow A_{i*}) \\ \epsilon & \text{otherwise} \end{cases} \tag{F3}$$

If the minimal hitting sequence *d-Ch* and $A_{i*}$ don't have same operations ($A_{i*} \cap$ $d$-$Chs_k = \emptyset$), Pr($A_{i*}$ | $d$-$Chs_k$) equals 0. If the operations of minimal hitting sequence *d-Ch* are exactly same with $A_{i*}$ ($A_{i*}$ = $d$-$Chs_k$), Pr($A_{i*}$ | $d$-$Chs_k$) equals 1. If neither, Pr($A_{i*}$ | $d$-$Chs_k$) equals ϵ , which is defined as (F4). The key idea undering our approach is that for each candidate $d$-$Chs_k$ we compute the g($d$-$Chs_k$) for the candidate's faulty operations that maximize the probability Pr($A_{i*}$ | $d$-$Chs_k$) of the matrix when r = 1,conditioned on that candidate $d$-$Chs_k$ (maximum likelihood estimation for naïve Bayes classifier $d$-$Chs_k$).

$$\epsilon = \begin{cases} \mathrm{g}\left(d-Chs_k\right)^{\mathrm{t}} & \text{if obtain a consistent ontology} \\ 1-\mathrm{g}\left(d-Chs_k\right)^{\mathrm{t}} & \text{if obtain an inconsistent ontology} \end{cases} \tag{F4}$$

Where t denotes the number of the same operations which are contains both the $i^{th}$ row $A_{i*}$ of the matrix and the minimal hitting sequence $d$-$Chs_k$. The parameter g($d$-$Chs_k$) is defined as formula (F5)

$$g(d-Chs_k) = \frac{\sum_{i=1}^{N}[(\bigcup_{j \in d-Chs_k} A_{ij} = 1) \wedge r_i = 0]}{\sum_{i=1}^{N}[\bigcup_{j \in d-Chs_k} A_{ij} = 1]} \tag{F5}$$

Before computing the posterior probability, we propose the computational methods of prior probability. If the cardinality of a hitting set *d-Chs*, |*d-Chs*| = 1, we set the prior

probability Pr($d$-$Chs_1$) = 0.01, which denotes the property that an operation causes the conflict. The computing formula of the other minimal hitting set is as formula (F6)

$$\Pr(d - Chs_k) = p^{|d-Chs_k|}(1-p)^{M-|d-Chs_k|} \tag{F6}$$

Where $M$ denotes the number of operations.

## 5. Instance Illustration

We practice the conflict diagnosis process for ontology evolution in opening environment once again by verifying an instance, with the premise is that the initial ontology $O$ is consistent. There are 5 changing sequences ($A_{1*}$,…$A_{5*}$) and 6 changing operations ($Och_1$, $Och_2$,…$Och_6$), which are encoded into a 5×6 matrix, and the last column $R$ represents the state of the result ontology $O'$, where $r_i$ = 0 ($O' \vDash COM$)和 $r_i$ = 1 ($O' \nvDash COM$).

$$\begin{array}{c} \\ A_{1*} \\ A_{2*} \\ A_{3*} \\ A_{4*} \\ A_{5*} \end{array} \begin{array}{c} \begin{array}{cccccc} Och_1 & Och_2 & Och_3 & Och_4 & Och_5 & Och_6 \end{array} \\ \left( \begin{array}{cccccc} 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{array} \right) \end{array} \begin{array}{c} R \\ \left( \begin{array}{c} 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{array} \right) \end{array} \tag{M3}$$

For the matrix above, there are two evaluations from users. The first one gets to stand by the second changing sequence $A_{2*}$ = <1, 1, 1, 0, 0, 0>. According to the algorithm 1 MUOUE((*A*,*R*), *u.eval*), we update the second row of the matrix, the result is shown in matrix (M6.4).

$$\begin{array}{c} \\ A_{1*} \\ A_{2*} \\ A_{3*} \\ A_{4*} \\ A_{5*} \end{array} \begin{array}{c} \begin{array}{cccccc} Och_1 & Och_2 & Och_3 & Och_4 & Och_5 & Och_6 \end{array} \\ \left( \begin{array}{cccccc} 0 & 1 & 0 & 1 & 1 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{array} \right) \end{array} \begin{array}{c} R \\ \left( \begin{array}{c} 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{array} \right) \end{array} \tag{M4}$$

Unlike any changing sequence that has existed in the matrix, the second evaluation contains two changing operations, which are $Och_1$ and $Och_6$ respectively. So we add the sequence as a new row $A_{6*}$ of the matrix. And because O($A_{6*}$)$\nvDash COM$, $r_6$ = 1.

$$\begin{array}{c} \\ A_{1*} \\ A_{2*} \\ A_{3*} \\ A_{4*} \\ A_{5*} \\ A_{6*} \end{array} \begin{array}{c} \begin{array}{cccccc} Och_1 & Och_2 & Och_3 & Och_4 & Och_5 & Och_6 \end{array} \\ \left( \begin{array}{cccccc} 0 & 1 & 0 & 1 & 1 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{array} \right) \end{array} \begin{array}{c} R \\ \left( \begin{array}{c} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{array} \right) \end{array} \tag{M5}$$

 

Below we will discuss the calculation of the minimal hitting sequence set using algorithm 2 CMHSS($A$,$R$).

(1) Firstly, based on the first loop function of algorithm 2 in line 2, the rows, whose results vector $r$=0, are deleted, the result matrix is shown in formula (M6).

$$\begin{array}{c} \\ A_{2*} \\ A_{3*} \\ A_{5*} \\ A_{6*} \end{array} \begin{array}{c} \begin{array}{cccccc} Och_1 & Och_2 & Och_3 & Och_4 & Och_5 & Och_6 \end{array} \\ \begin{pmatrix} 2 & 2 & 2 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \begin{array}{c} R \\ \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \end{array} \quad \text{(M6)}$$

(2)According to the second loop function, we check whether the solution exists. As noted in the Corollary 3, existence of solution is proved.

(3)According to the third loop function, we go through each column of the matrix, to check that whether there are some columns all elements of which are 0. As is mentioned in Corollary 2, the corresponding operations will be not contained in the minimal hitting sequence set. So we delete these columns. As it is clearly that the fifth column of (M6) needs to be deleted, and the result is as (M7).

$$\begin{array}{c} \\ A_{2*} \\ A_{3*} \\ A_{5*} \\ A_{6*} \end{array} \begin{array}{c} \begin{array}{ccccc} Och_1 & Och_2 & Och_3 & Och_4 & Och_6 \end{array} \\ \begin{pmatrix} 2 & 2 & 2 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \begin{array}{c} R \\ \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \end{array} \quad \text{(M7)}$$

(4)The fourth loop function of algorithm 2 is the detection that whether there are some operations which are included by all conflict changing sequences. If there are some, according to Corollary 1, each of the operations can be as a minimal hitting sequence which contains only one operation. We can see that there are no such operations.

(5)We work out the number of times of each operation appeared in the matrix through formula (F1). The result is shown in (M8). And according to the line 6.a) of algorithm 2, $m_j$ = 4, the corresponding changing operation is $Och_1$.

$$\begin{array}{c} \\ A_{2*} \\ A_{3*} \\ A_{5*} \\ A_{6*} \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccc} Och_1 & Och_2 & Och_3 & Och_4 & Och_6 \end{array} \\ \begin{pmatrix} 2 & 2 & 2 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 4 & 3 & 3 & 1 & 1 \end{pmatrix} \end{array} \begin{array}{c} R \\ \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ \\ \end{pmatrix} \end{array} \quad \text{(M8)}$$

Then we break the matrix by $Och_1$. We delete the rows which contain the changing operation $Och_1$. The new matrix is as (M9).

$$\begin{array}{c} \quad Och_2 \quad Och_3 \quad Och_4 \quad Och_6 \quad R \\ \hline A_{5^*}\begin{pmatrix} 1 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix}1\end{pmatrix} \\ \quad 1 \quad 0 \quad 1 \quad 0 \end{array} \tag{M9}$$

①The algorithm is recalled with the new matrix (M9) as input. Then the columns, whose elements are all equal to 0, are deleted and get the result matrix in (M10).

$$\begin{array}{c} \quad Och_2 \quad Och_4 \quad R \\ \hline A_{5^*}\begin{pmatrix} 1 & 1 \end{pmatrix}\begin{pmatrix}1\end{pmatrix} \\ \quad 1 \quad 1 \end{array} \tag{M10}$$

②For $Och_2$ and $Och_4$ have the same number of times, and all elements of their corresponding columns are equal to 1, so the minimal hitting sequence set of the decomposition matrix based on $Och_1$ is {{ $Och_2$}, { $Och_4$}}.

⑹Then the column corresponding changing operation $Och_1$is deleted from the matrix (M7). And we can get the matrix like (M 11).

$$\begin{array}{c} \quad Och_2 \quad Och_3 \quad Och_4 \quad Och_6 \quad R \\ \hline \begin{matrix} A_{2^*} \\ A_{3^*} \\ A_{5^*} \\ A_{6^*} \end{matrix}\begin{pmatrix} 2 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}\begin{pmatrix}1\\1\\1\\1\end{pmatrix} \\ \quad 3 \quad 3 \quad 1 \quad 1 \end{array} \tag{M11}$$

① For $Och_2$ and $Och_4$ have an equal number of times, we take $Och_2$ as the basis of matrix decomposition in random, and then the matrix was decomposed.

$$\begin{array}{c} \quad Och_3 Och_4 Och_6 \; R \\ \hline \begin{matrix} A_{3^*} \\ A_{6^*} \end{matrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix} \\ \quad 1 \quad 0 \quad 1 \end{array} \tag{M12}$$

②Further, the second column is deleted for all its elements are equal to 0. The matrix turns into (M 13).

$$\begin{array}{c} \quad Och_3 \quad Och_6 \; R \\ \hline \begin{matrix} A_{3^*} \\ A_{6^*} \end{matrix}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix} \\ \quad 1 \quad 1 \end{array} \tag{M13}$$

③The changing operation $Och_3$ is taken as the standards to decompose the matrix, and obtaining a result matrix (M 14).

$$\begin{array}{c} \quad Och_6 \; R \\ \hline A_{6^*}(1)(1) \\ \quad 1 \end{array} \tag{M14}$$

 

The minimal hitting sequence set is { $d\text{-}Chs_3$ = { $Och_2$, $Och_3$, $Och_6$ }}.

⑺Similarly, we take $Och_3$ as the basis of matrix decomposition in random, and then the matrix (M11) was decomposed. Then, we continue the recursive call. And eventually get the minimal hitting sequence set { $d\text{-}Chs_4$ = { $Och_2$, $Och_3$, $Och_4$, $Och_6$ }}. For $d\text{-}Chs_3 \subset d\text{-}Chs_4$, $d\text{-}Chs_4$ is given up.

⑻Then $Och_2$ and $Och_3$ are deleted from matrix (M11), the minimal hitting sequence set is { $d\text{-}Chs_4$ = { $Och_2$, $Och_3$, $Och_4$, $Och_6$ }}. For the same reason, $d\text{-}Chs_3 \subset d\text{-}Chs_4$, $d\text{-}Chs_4$ is given up, too.

⑼So $D\text{-}Chs$ = { $d\text{-}Chs_1$ = { $Och_1$, $Och_2$}, $d\text{-}Chs_2$= { $Och_1$, $Och_4$}, $d\text{-}Chs_3$ = { $Och_2$, $Och_3$, $Och_6$ }}.

⑽We use Bayes formula to ranking the minimal hitting sequences of the step ⑼. We present our approach to compute the posterior candidate properties Pr($d\text{-}Chs_k$).

①Firstly, we calculate the numerator of each minimal hitting sequence set. The formula is (F3). There are three conditions within the formula. The first one is that $d\text{-}Ch$ and $A_{i*}$ don't have same operations ($A_{i*} \cap d\text{-}Chs_k = \emptyset$), then the numerator is 0. The second one is that the operations of minimal hitting sequence $d\text{-}Ch$ are exactly same with $A_{i*}$ ($A_{i*} = d\text{-}Chs_k$), in this case the numerator is 1. Our example dose not meet both the two cases. The formula (F4) is used to compute ϵ. In our example, the computation of numerator of the first minimal hitting sequence $d\text{-}Chs_1$ is as (M15).

$$\begin{array}{c|cccccc|c|c} & Och_1 & Och_2 & Och_3 & Och_4 & Och_5 & Och_6 & R & \Pr(A_{i*} \mid d-Chs_1) \\ \hline A_{1*} & 0 & 1 & 0 & 1 & 1 & 0 & 0 & g(d-Chs_1)^1 \\ A_{2*} & 2 & 2 & 2 & 0 & 0 & 0 & 1 & 1-g(d-Chs_1)^2 \\ A_{3*} & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1-g(d-Chs_1)^1 \\ A_{4*} & 0 & 1 & 0 & 1 & 0 & 1 & 0 & g(d-Chs_1)^1 \\ A_{5*} & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1-g(d-Chs_1)^1 \\ A_{6*} & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1-g(d-Chs_1)^1 \end{array} \quad \text{(M15)}$$

Where $t$ is the number of intersection of the changing sequences in the matrix and the minimal hitting sequence. The result of a computation is shown in (M16).

$$\begin{array}{c|cccccc|c|c|c} & Och_1 & Och_2 & Och_3 & Och_4 & Och_5 & Och_6 & R & \Pr(A_{i*} \mid d-Chs_1) & t \mid d-Chs_1 \\ \hline A_{1*} & 0 & 1 & 0 & 1 & 1 & 0 & 0 & g(d-Chs_1)^1 & 1 \\ A_{2*} & 2 & 2 & 2 & 0 & 0 & 0 & 1 & 1-g(d-Chs_1)^2 & 2 \\ A_{3*} & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1-g(d-Chs_1)^1 & 1 \\ A_{4*} & 0 & 1 & 0 & 1 & 0 & 1 & 0 & g(d-Chs_1)^1 & 1 \\ A_{5*} & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1-g(d-Chs_1)^1 & 1 \\ A_{6*} & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1-g(d-Chs_1)^1 & 1 \end{array}$$

(M16)

By the formula (F5), g($d\text{-}Chs_1$) = 2/9.

$$\begin{array}{c|cccccc|c|c|c} & Och_1 & Och_2 & Och_3 & Och_4 & Och_5 & Och_6 & R & \Pr(A_{i*} \mid d-Chs_1) & t \mid d-Chs_1 \\ A_{1*} & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 2/9 & 1 \\ A_{2*} & 2 & 2 & 2 & 0 & 0 & 0 & 1 & 1-(2/9)^2 & 2 \\ A_{3*} & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1-2/9 & 1 \\ A_{4*} & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 2/9 & 1 \\ A_{5*} & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1-2/9 & 1 \\ A_{6*} & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1-2/9 & 1 \end{array} \quad \text{(M17)}$$

②$\Pr(A_{i*})$ is a normalization factor. So there are no effects on the ranking result.

③There is a formula (F6) for calculating priori probability, where we set $p$ = 0.01. In the case that operations are separate non-interacting, and because | $d$-$Chs_1$| = 2, the priori probability of $d$-$Chs_1$ can be obtained as $\Pr(d\text{-}Chs_1) = 0.01^2*(1\text{-}0.01)^4$.

④Based on above data, the posterior probability of $d$-$Chs_1$ can be computed as (M18).

$$\begin{array}{c|c|c} & R & \Pr\left(d-Chs_1 \mid A_{i*}\right) \\ A_{1*} & 0 & 2/9\times 0.01^2\times(1-0.01)^4 \\ A_{2*} & 1 & [1-(2/9)^2]\times \Pr\left(d-Chs_1 \mid A_{i1}\right)/2 \\ A_{3*} & 1 & (1-2/9)\times \Pr\left(d-Chs_1 \mid A_{i2}\right)/3 \\ A_{4*} & 0 & 2/9\times \Pr\left(d-Chs_1 \mid A_{i3}\right)/4 \\ A_{5*} & 1 & (1-2/9)\times \Pr\left(d-Chs_1 \mid A_{i4}\right)/5 \\ A_{6*} & 1 & (1-2/9)\times \Pr\left(d-Chs_1 \mid A_{i5}\right)/6 \end{array} \quad \text{(M18)}$$

⑤The posterior probabilities' calculate process of two other candidates are similar. The results is listed in (M19).

$$\begin{array}{c|c} d-Chs_k & \Pr\left(d-Chs_k \mid A_{i*}\right) \\ d-Chs_1 & 0.046 \\ d-Chs_2 & 0.105 \\ d-Chs_3 & 2.52E-5 \end{array} \quad \text{(M19)}$$

So the final diagnosis ranking is $\Pr(d\text{-}Chs_2|A_{i*}) > \Pr(d\text{-}Chs_1|A_{i*}) > \Pr(d\text{-}Chs_3|A_{i*})$.

## 6. Conclusions and Future Work

For the characteristics of Gene Ontology Evolution in the opening environment, in this paper, we present a dynamic diagnosing conflicts technique, which is based on the changing sequence matrix. The changing sequences of evolution is modeled as a matrix. We consider the characters of the opening environment, and propose a updating algorithm to update the matrix. Our diagnosis approach can provide an explanation for conflicts of the changing sequences by ranking the minimal hitting sequence set.

 

## Acknowledgements

This work is supported by the The PhD Start-up Fund of ShanDong Institute of Business and Technology (DS201405) and the Open Project Program of Artificial Intelligence Key Laboratory of Sichuan Provence(Sichuan University of Science and Engineering), China(2013RYJ05).

## References

[1] A. Burgun, G. Botti and P. Le Beux, Syst., vol. 25, **(2001)**, pp. 95-108.
[2] H. Singh, C. Roman, O. Pizarro, R. Eustice and A. Can, Int. J. Rob. Res. vol. 26, **(2007)** pp. 55-74.
[3] W. Kuśnierczyk, "Taxonomy-based partitioning of the Gene Ontology", J. of Biomedical Informatics vol. 41, **(2008)**, pp. 282-292.
[4] M. Sikora and A. Gruca, "Pattern Recogn. Lett, vol. 32, **(2011)**, pp. 258-269.
[5] A. Morgan, L. Hirschman, A. Yeh and M. Colosimo, "Proceedings of the ACL 2003 workshop on Natural language processing in biomedicine", Association for Computational Linguistics, Sapporo, Japan, vol. 13, **(2003)**, pp. 1-8.
[6] L. F. B. Seibel and S. Lifschitz, "A genome databases framework", Database and Expert Systems Applications, Springer, **(2001)**, pp. 319-329.
[7] D. Qi, J. A. Blake, J. A. Kadin, J. E. Richardson, M. Ringwald, J. E. Eppig and C. J. Bult, Proceedings of the 2005 IEEE Computational Systems Bioinformatics Conference – Workshops, IEEE Computer Society, **(2005)**, pp. 37-38.
[8] G. Fitzpatrick, P. Marshall and A. Phillips, Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work, ACM, Banff, Alberta, Canada, **(2006)**, pp. 49-58.
[9] L. Voinea and A. Telea, Proceedings of the 2006 international workshop on Mining software repositories, ACM, Shanghai, China, **(2006)**, pp. 33-39.
[10] C. Tempich, H. Pinto, Y. Sure and S. Staab, "The Semantic Web: Research and Applications", **(2005)**, pp. 21-43.
[11] Y. Chen, X. Peng and W. Zhao, "Advances in Data and Web Management, **(2009)**, pp. 442-454.
[12] R. Palma, O. Corcho, A. Gómez-Pérez and P. Haase, "Web Semantics: Science, Services and Agents on the World Wide Web", vol.9 , **(2011)**, pp. 299-314.
[13] S. Zhong and D. Xie, Artif. Intell. Med, vol. 41, **(2007)** , pp. 105-115.
[14] K. Seki and J. Mostafa, Inf. Process. Manage. vol. 44, **(2008)**, pp.1754-1770.
[15] F. Saglietti, Proceedings of the fifth Jerusalem conference on Information technology, IEEE Computer Society Press, Jerusalem, Israel, **(1990)**, pp. 270-277.
[16] T. J. Ostrand, E. J. Weyuker and R. M. Bell, IEEE Trans. Softw. Eng, vol.31, **(2005)**, pp. 340-355.
[17] R. Abreu and A. J. C. V. Gemund, Artif. Intell, vol. 174, **(2010)**, pp. 1481-1497.
[18] R. Abreu, P. Zoeteweij and A. J. C. V. Gemund, "J. Syst. Softw. vol.84, **(2011)**, pp.573-586.
[19] Y. Song, R. Chen, H. Papadopoulos, A. Andreou and M. Bramer, "Artificial Intelligence Applications and Innovations", IFIP Advances in Information and Communication Technology, Springer Boston, vol. 339, **(2010)** , pp. 203-210.
[20] R. C. Y. Song and Y. Liu, Journal of Computers, vol.7, **(2012)**, pp. 2454-2461.
[21] W. L. Hürsch, Proceedings of the ACM Conference on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA'97), ACM SIGPLAN Notices, vol. 32, **(1995)**.

## Authors

**Qiu Ling**

2010 Master's degree in University of Electronic Science and Technology of China

2002 Bachelor's degree in Sichuan University of Science & Engineering

**Research interests** Computer Application, Artificial Intelligence

**Liu Yong**

2003~current Research at Artificial Intelligence Key Laboratory of Sichuan Provence

2003 Bachelor's degree in Sichuan University of Science & Engineering

**Research interests** Semantic Web, Artificial Intelligence

**Song Yingjie**

2013 Doctor's degree in Dalian Maritime University

2009 Master's degree in Dalian Maritime University

**Research interests** Semantic Web, Ontology Evolution, Behavior Identification

**Zhang Bin**

2010 Master's degree in Dalian Maritime University

2008 Bachelor's degree in Qilu University of Technology

**Research interests** Software Quality Assurance、Embed-Software testing、Software reliability testing