Computer Standards & Interfaces 41 (2015) 10–16

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

A model for tracing cross-referenced statement validity

Jinsongdi Yu

Spatial Information Research Center (SIRC), Fuzhou University, 350002 Fuzhou, China

ARTICLE INFO

Article history:
Received 26 November 2013
Received in revised form 15 November 2014
Accepted 8 February 2015
Available online 24 February 2015

Keywords:
Interoperability
Conformance testing
Global validity
Fixed point

ABSTRACT

Big Data Era brings global digital infrastructure collaboration built on the emerging standards. Given the complexity and dynamics of each specification, corresponding implementations need to undergo sufficient verification and validation procedures. Significant efforts have been invested into conformance testing of individual requirements, for example, by using formal, semi-formal or informal approaches. Less works have been accomplished, however, on the overall orchestration assessment so as to ensure global validity of conformance statements. For example, cyclic dependencies among conformance statements of a service under test may lead to inappropriate conclusions on the assessment outcome. In this study, a dependency model based on three-valued logic and fixed point theory to address dependency issues among cross-referenced statements is presented, so as to provide effective support to global digital infrastructure collaboration.

1. Introduction

In Big Data Era, traditional computer-based communication becomes more widespread and users begin to experiment with newer sensor-based communication, such as Imote2 sensor networks and Geo-Processing Workflows. To improve interoperability and archive a better global digital infrastructure collaboration. A variety of standards on networking, human–computer interaction, artificial intelligence and etc. are produced in varying degrees of maturity and maintained or supported by standardization bodies, such as IEEE, ISO, W3C, OGC, and IETF. These specifications may complement, overlap, and compete with each other. A conformance statement asserts the conformance of a specific requirement by testing a given service component implementation. It is often authored as a result of the related conformance testing. A potential user of the implementation can consult the conformance statements to determine if it meets his or her requirements. However, the complexity of web services and the interrelationships and dependencies among the standards establishes additional problems; one of them, which is addressed in this work, is related to dependencies between conformance statements. Such dependencies often exist already within one specification against which a service is to be assessed. Moreover, specifications themselves often refer to other specifications — either because some pre-existing specification is used, like in the case of relying on standards, or because the specification on hand is modularized into different documents between which, consequently, logical relationships exist. Three similar dependency relationships are defined by Belguidoum and Dagnat [8] with mandatory, optional or negative forms. However, the work does not mention the overall orchestration assessment so as to ensure global validity of cross-referenced statements. Obviously, a statement can only be fully trusted if its dependencies are known to be true; otherwise the orchestration assessment of these statements would lead to inappropriate conclusions.

Following a three-valued logic, three possible statement statuses, namely T, F, and U, are distinguished. The statement state is *unknown* before testing and *true*, *false*, or *unknown* after test execution. Accordingly, statement in three-valued logic (STL) is proposed to address dependency issues among these statements; among the goals is to allow collecting feedback for improving specification, test suites and implementations. The remaining parts of this paper are organized as follows: Section 2 introduces related research, Section 3 provides the formalized dependency model based on three-valued logic expressions, Section 4 proves the global validity evaluation approach, and Section 5 summarizes this paper and discusses the future.

2. State of the art

Traditionally, conformance testing investigates whether a product or system adheres to properties defined, for example, in some standards. Two-valued logic provides truth values indicating true and false results. On the overall orchestration of conformance statements so as to ensure global validity of the results, traditional schedule approaches in graph theory or Dependency Structure Matrix [27] (DSM) help in mapping the dependency relationships. The graph traversal helps in evaluating the statement in proper order. Among them, several works have been done on ordering the cycles in none-singleton strongly connected components (NSSs). For example, Kung et al. [22] selects a random dependency to break cycles, and Le Traon et al. [23], Tai and Daniel [28], Hewett et al. [20] and Briand et al. [9] deploy their removal strategies according to the number of incoming edges and the number of

E-mail address: yyx350@gmail.com.

http://dx.doi.org/10.1016/j.csi.2015.02.005
0920-5489/

outgoing edges. Kraft et al. [21] remove the edges according to its dependency weight function. However, these serializations lose the dependency information of the removal nodes. These may introduce inappropriate conclusions in the orchestration assessment of the global validity of cross-referenced statements. Similar dependency analysis [14] is studied based on truth maintenance systems (TMS) in the 80s and early 90s, for instance by Goodwin [18,19], to establish the stability of such networks. These approaches are based on Boolean TMSs and negation is included. However, the cases in which logically dependent relationships occur in three-valued logic are not considered. For example, Kleene's three-valued logic [15,16] provides such expressiveness by adding a third value for a specific semantic other than true or false. Although a map of such dependency relationships between conjunctions, negations and implications that extend the Boolean one is presented [17], how to address the corresponding evaluation schedule containing strongly connected components (SCC) remains open. Without proper evaluation schedules, the results tend to be undecidable.

3. The dependency model in three-valued logic

3.1. Statement in three-valued logic

Logical *conjunction* ("and") and *disjunction* ("or") operators are used to distinguish statements. An *atomic statement* (AS) is a logic statement which cannot be broken down into smaller statements; a *composite statement* (CS) is a logic statement having two or more statements connected by logical *conjunction* ("∧") and *disjunction* ("∨") operators. Syntactically, a logical expression is represented by this grammar:

$$\begin{aligned} AS &: \text{"T"}|\text{"F"}|\text{"U"} \\ OP &: \text{"}\wedge\text{"}|\text{"}\vee\text{"} \\ S &: AS|S\,OP\,S|\text{"("}S\text{")"}. \end{aligned}$$

Kleene's logic [16] is used to evaluate these expressions. In Kleene's logic, a conjunction produces a value of T if both of its operands are T, an F if one of its operands is F, and otherwise U. Disjunction delivers a value of T if one of its operands is T, an F if both of its operands are F, and otherwise a U.

Kleene uses the Open World Assumption (OWA) approach to evaluate logic expressions [15]. OWA states that the truth value of a statement that is not included in itself or inferred from the knowledge explicitly recorded in the expression shall be considered unknown.

Besides the evaluation in OWA, Closed World Assumption (CWA) has already been used a lot to evaluate knowledge representation statements within some given system [11]. CWA is the assumption that any statement that is not known to be true is false. Yet another case in software testing is distinguished: frequently there are test stubs which simulate the behavior of the dependent test modules. In this case, the dependent test modules are always assumed to be true when their truth evaluation results are not available. This is called a Stub Assumption (SA); the dependent statement is true under a SA.

Three assumptions are distinguished on unknown features (Table 1): Open World Assumption (OWA), Closed World Assumption (CWA), and Stub Assumption (SA). Accordingly, the truth table of the logic operations for Kleene's logic is extended with CWA and SA, see Table 2.

Table 1
Assumptions on unknown results.

Assumption	Operand	Result
CWA	U	F
OWA	U	U
SA	U	T

Table 2
Extend the truth table of the logic operations for Kleene's logic with CWA and SA.

P	Q	$P \wedge Q$ (CWA)	$P \wedge Q$ (Kleene's logic, OWA)	$P \wedge Q$ (SA)	$P \vee Q$ (CWA)	$P \vee Q$ (Kleene's logic, OWA)	$P \vee Q$ (SA)
T	*T*	*T*	*T*	*T*	*T*	*T*	*T*
T	*U*	*F*	*U*	*T*	*T*	*T*	*T*
T	*F*	*F*	*F*	*F*	*T*	*T*	*T*
U	*T*	*F*	*U*	*T*	*T*	*T*	*T*
U	*U*	*F*	*U*	*T*	*F*	*U*	*T*
U	*F*	*F*	*F*	*F*	*F*	*U*	*T*
F	*T*	*F*	*F*	*F*	*T*	*T*	*T*
F	*U*	*F*	*F*	*F*	*F*	*U*	*T*
F	*F*	*F*	*F*	*F*	*F*	*F*	*F*

3.2. Dependency model

The model based on the directed graph is developed for tracing dependencies among statements. The model starts with a directed dependency graph $G = (V,E)$. The vertex set V contains statements as vertices, the edge set E consists of dependencies; an edge $e = (s_1,s_2)$ denotes that vertex s_1 depends on vertex s_2. s_2 is said to be a direct successor of s_1 while s_1 is said to be a direct predecessor of s_2. Generally, if a path in G is made up one or more successive edges leading from vertex s_1 to vertex s_2, then s_2 is said to be a successor of s_1 and s_1 is said to be a predecessor of s_2. If there is a path from s_1 to s_2 and also a return path from s_2 to s_1, then s_1 and s_2 are strongly connected. A strongly-connected component (SCC) [10] is a maximal strongly-connected subgraph of G. SCC decomposition is a partition of the set of vertices.

A statement is evaluated according to its direct successors. The form is listed as below:

a. $P \rightarrow Q$
b. P (Hypothesis stated)
c. Q (Conclusion given).

For example, statement s_1 depends on s_2 and s_3 and (s_4 or s_5), its corresponding dependencies are shown in Fig. 1. In this case, s_2 and s_3 and (s_4 or s_5) is P and s_1 is Q.

3.3. Deadlock and false deadlock

Dependencies among statements may introduce deadlocks. For example, let us assume that statement s_1 depends on s_2 and s_3, s_3 depends on s_4 and s_4 depends on s_1. This means that s_1, s_3 and s_4 are strongly-connected as shown in Fig. 2.

The cycle introduces a deadlock in the evaluation. However, not all the deadlock is completely dead.

A node's truth value may be determined by parts of its dependencies. For example, $F \wedge exp$ will always return F even if the truth value of exp is unknown. If such a node is in a circle the node can be disentangled. For example, see Fig. 2, if $s_2 = F$, the rest truth values can be derived. This deadlock is a *false deadlock*.

A possible truth list (PTL) is used to represent the list of possible truth values for a given statement. A possible truth result expression (PTRE) is a logical expression having two or more PTLs connected by logical *conjunction* ("∧") and *disjunction* ("∨") operators. Syntactically, the grammar is given as below:

$$\begin{aligned} &PTL : \text{"}T\text{"}|\text{"}U\text{"}|\text{"}F\text{"}|\text{"}T,U\text{"}|\text{"}U,F\text{"}|\text{"}T,U,F\text{"}|\text{"}T,F\text{"}|\text{"("PTL")"} \\ &CDOP : \text{"}\wedge\text{"}|\text{"}\vee\text{"} \\ &PTRE : PTL|\text{"("PTRE")"}CDOP\text{"("PTRE")"}. \end{aligned}$$

If the possible truth list cardinality $l = 1$, the truth value of the node can be derived directly. A *false deadlock* contains at least one such node. Below are the PTL truth tables under open world assumption (see