Measure Confidence of Assurance Cases in Safety-Critical Domains

Chung-Ling Lin, Wuwei Shen
Department of Computer Science
Western Michigan University
Kalamazoo, MI, USA
{chung-ling.lin,
wuwei.shen}@wmich.edu

Steven Drager
Air Force Research Laboratory
Rome, NY, USA
steven.drager@us.af.mil

Betty Cheng
Department of Computer Science & Engineering
Michigan State University
East Lansing, MI, USA
chengb@cse.msu.edu

ABSTRACT

Evaluation of assurance cases typically requires certifiers' domain knowledge and experience, and, as such, most software certification has been conducted manually. Given the advancement in uncertainty theories and software traceability, we envision that these technologies can synergistically be combined and leveraged to offer some degree of automation to improve the certifiers' capability to perform software certification. To this end, we present a novel confidence calculation framework that 1) applies the Dempster-Shafer theory as a mathematical model to calculate the confidence between a parent claim and its children claims; and 2) uses the vector space model to evaluate the confidence for the evidence items using traceability information. A fragment of an assurance case (expressed in the goal-structuring notation – GSN) for the coupled tank system is used to illustrate our new framework.

KEYWORDS

Software Certification, Dempster-Shafer theory, Software Traceability, Vector Space Model (VSM)

1 INTRODUCTION

Evaluation of assurance cases requires certifiers to have extensive domain knowledge and experience. As such, most software certification, including that used in the safety-critical sectors, has been conducted manually. Due to the growing complexity of modern software, however, assurance cases are increasingly exponentially in size and complexity. For example, the preliminary safety case for co-operative airport surface surveillance operations is approximately 200 pages long [1], where the size is expected to grow as more operations are considered. Manual certification is thus not only time consuming but also error prone and expensive.

With the success of the Dempster-Shafter (D-S) theory applied in different domains in order to manage uncertainty for decision-making (e.g., accident prediction, landslide management, etc.), researchers have employed it to evaluate an assurance case. Current D-S based approaches [2], however, require human involvement to assign values to assessment parameters such as a disjoint contributing weight of a child node, indicating its appropriateness of the child node independently contributing to the belief of its parent node in an assurance case when propagating the confidence information from child nodes to their parent node.

Recently, safety case patterns have been employed to generate new assurance cases [3, 4] using the Goal Structuring Notation (GSN) [5], thus suggesting that assurance cases have recurring structures especially in a large assurance case. This trend has motivated us to explore techniques to automate portions of the software certification process. This paper introduces a new framework to synergistically combine the D-S theory with the Vector Space Model (VSM) [7] in order to automate the assurance case confidence calculation. Specifically, the D-S theory is used to propagate the confidence information from leaf claims up towards the root claim of an assurance case. And VSM that is used to initially deduces confidence information for leaf claims, is then used to maximally automate the update of the confidence calculation without human intervention.

Specifically, the framework takes as input an acceptable assurance case for a large software system. This assurance case can be produced as a claim on one activity during a software development lifecycle (SDLC). From the input, the framework initially generates a set of similar assurance cases, each of which has the same structure as the input assurance case except for the leaf claims that are directly supported by an evidence node in GSN. We further notice that confidence of many leaf claims can be achieved by checking the similarity made in its supporting evidence node. As such, the framework employs the VSM to deduce confidence values for leaf claims. When using the D-S theory to propagate confidence from leaf claims to a root claim, the framework automatically calculates values for all disjoint contributing weights by making the acceptable assurance case to have a higher rank than most generated assurance case in terms of

ICSE-NIER'18, May 27-June 3, 2018, Gothenburg, Sweden
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5662-6/18/05…$15.00
https://doi.org/10.1145/3183399.3183419

confidence calculation. Next, the framework applies the derived values for the disjoint contributing weights to the second assurance case, that can claim on another activity of the SDLC but has the same structure as the initially-certified assurance case, and thus deduces a confidence value on the second assurance case.

We apply the framework to the Coupled Tank System developed by AFRL to address the coupled tanks challenge problem [6]. Our preliminary results show that the framework can successfully evaluate an unknown assurance case based on an initial acceptable assurance case.

2 PROPOSED FRAMEWORK

We start with an overview of an industrial-strength application used as a running example. Then we overview the framework, including each step for the confidence calculation approach.

2.1 Case Study: Coupled Tank System

The Coupled Tank challenge system draws liquid from a limitless source, temporarily store the liquid for some process such as missing or temperature control, and finally releases the liquid into a bottomless sink. The requirements phase starts with the concept of operations (CONOPS) that denotes a set of high-level requirements for system, where the AFRL team derives a set of system requirements based on three aspects: the system, controller and environment. Then the specification and analysis of requirements (SpeAR) framework is used to develop and analyze the system requirements and is called an analysis phase. Namely, the system requirements are further decomposed into SpeAR properties and/or requirements respectively. In this case, after applying the same safety pattern, Figure 1 (i) and (ii) show the two assurance cases that claims about the two phases respectively. To support the top claim, rendered as a rectangle, in the coupled tank system (*i.e.*, 0G1.1.1), both assurance cases employ three sub-claims, *i.e.*, 0G1.2.1, 0G1.2.2, and 0G1.2.3, that refer to the system, environment, and controller aspects respectively. A strategy node, rendered as a parallelogram, *e.g.,* 0S1.1.1, represents how a parent claim is supported by its sub-claims. The support relation in GSN is given by a line ending with an arrow. Next, for each sub-claim 0G1.2.x, two sub-claims are further developed to support its correctness and completeness. For instance, for the "*Requirements at the System Aspect are adequately elicited and documented in the requirement document*" sub-claim node in Figure 1 (i), the correctness sub-claim node claims that all system requirements at the system aspect correctly implement the high-level requirement (CONOPS). The completeness sub-claim node asserts that all high-level requirements are completely considered by the system requirements at the system aspect. Likewise, in the second assurance case shown in Figure 1(ii), for the "*SpeAR model properties at the System Aspect are adequately elicited and documented in the SpeAR model document"* sub-claim node, term "*SpeAR model properties*" replaces term "*Requirements*" and term "*SpeAR model document*" replaces term "*requirement document*" in Figure 1(i). Furthermore, each sub-claim 0G1.2.x in Figure 1(ii) is further supported by two sub-claims in terms of correctness and completeness as its counterpart in Figure 1(i). The difference between the two assurance cases is that 5 leaf claims are under 0S.1.6.1 in Figure 1(i) but 8 leaf claims in Figure 1(ii). The reason is that the two assurance cases consider 5 CONOPS requirements and 8 system requirements respectively.

2.2 Overview of the Framework

To calculate the confidence of a second assurance case such as that shown in Figure 1(ii), where the first assurance case (in Figure 1(i)) has been determined to be acceptable, the framework carries out the following steps. First, convert an assurance case into a confidence calculation model where D-S theory can be applied. Second, use Vector Space Model to calculate similarity values as confidence for all leaf nodes in a confidence calculation model. Third, apply the D-S theory to infer all the disjoint contributing weights in a common structure of two input assurance cases, using the fact that the first assurance case is acceptable. Fourth, apply all the disjoint contributing weights in the second assurance case to deduce its confidence.

2.3 Confidence Calculation Model and Vector Space Model

First, a confidence calculation model removes auxiliary nodes, such as a strategy node, justification node, and context node, to only include claim nodes and evidence nodes. Furthermore, if a claim node, called c_1 is supported by only one sub-claim c_2 that is further supported by sub-claim c_3, then we have claim node c_1 directly supported by sub-claim c_3 without sub-claim c_2 in a confidence calculation model. For instance, the confidence calculation model in Figure 2 is converted from the assurance case shown in Figure 1(i).

Confidence evaluation of an assurance case starts with leaf nodes, *i.e.*, evidence nodes, in its corresponding confidence calculation model. We notice that many assurance cases have traceability information at the leaf node level. Therefore, instead of asking for a confidence value as input, we use the Generalized Vector Space Model (GVSM) [7] to calculate a similarity value to mimic how a certifier evaluates a leaf node.

2.4 Application of the D-S Theory in Assurance Cases

We largely follow Wang et al's formulation for the D-S theory calculation [2]. In a calculation model, confidence of a claim node is represented by two formats. One is called the trustworthiness of a node, denoted as a 3-tuple (*bel*(P), *dis*(P), *uncer*(P)) showing belief, disbelief, and uncertainty of node P respectively. This format is used when applying the D-S theory. A frame of discernment Ω_P is $\{P, \bar{P}\}$, where $\bar{P}$ denotes false of P. The mass $m^{\Omega_P}(P)$ shows the degree of belief committed to the hypothesis that truth lies in P [2]. The 3-tuple is thus defined as follows:

$$\begin{cases} bel(P) = m^{\Omega_P}(P) = g_B \\ dis(P) = bel(\bar{P}) = m^{\Omega_P}(\bar{P}) = f_B \\ uncer(P) = m^{\Omega_P}(\Omega_P) = 1 - m^{\Omega_P}(P) - m^{\Omega_P}(\bar{P}) = 1 - g_B - f_B \end{cases} \quad (1)$$

Another format of a node represents a certifier's evaluation on node P using a 2-tuple (*dec(P)*, *conf(P)*) where *dec* denotes a

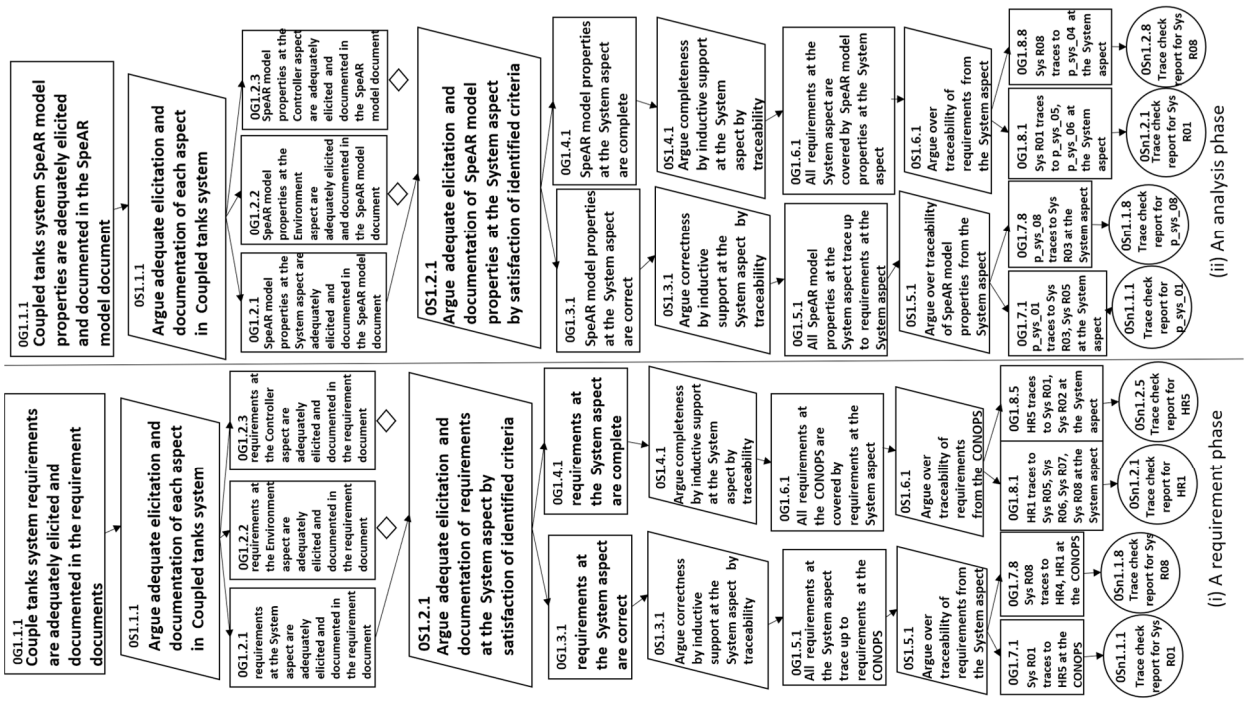

Figure 1: Two assurance cases for the coupled tank system

certifier's confidence value and *conf* represents the confidence about the method used by a certifier to have a value for *dec*. Some conversion is necessary. For instance, a certifier's evaluation on a leaf node via a 2-tuple is converted to a 3-tuple so the D-S theory can be carried out. Likewise, a 3-tuple of a root claim is converted to a 2-tuple after the calculation, mimicking a certifier's evaluation. Formulas (2) and (3) show the conversions between a 3-tuple and a 2-tuple of a node.

$$\begin{cases} bel(P) = conf(P) \times dec(P) \\ dis(P) = conf(P) \times (1 - dec(P)) \\ uncer(P) = 1 - bel(P) - disb(P) \end{cases} \quad (2)$$

$$\begin{cases} conf(P) = bel(P) + disb(P) \\ dec(P) = bel(P) \,/\, (bel(P) + disb(P)), \ if\ bel(P) + disb(P) \neq 0 \\ dec(P) = 0, \quad if\ bel(P) + disb(P) = 0 \end{cases} \quad (3)$$

The confidence calculation of a claim depends on its sub-claim(s). Two types of sub-claim contributions can be used to support a parent claim, *i.e.*, a dependent argument and redundant argument. A dependent argument is the contribution of one sub-claim to its parent claim depends on another (*i.e.*, sibling) sub-claim while a dependent argument denotes the contribution of a sub-claim to support its parent claim has some degree of overlap with another sub-claim. Next, assume that two sub-claims, *e.g.*, B and C, support a parent claim denoted as A. To represent the appropriateness of B and C to support A, we use w_B and w_C that are called disjoint contributing weights of B and C respectively. Since it is not possible to infer that the trustworthiness of a node can be derived from all its children nodes via the appropriateness, we use discounting factor v to denote the uncertainty about the appropriateness of all children claims to support their parent claim.

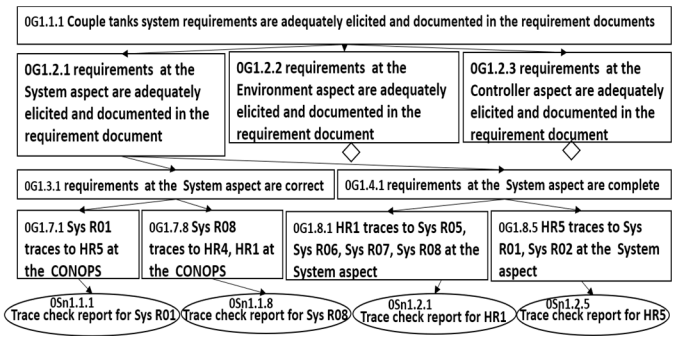

Figure 2 Requirement phase confidence calculation model

Next, the trustworthiness of a claim denoted as A with an n-claims dependent argument is given as follows:

$$\begin{cases} bel(A) = v[\left(1 - \sum_{i=1}^{n} w_i\right) \prod_{i=1}^{n} g_i + \sum_{i=1}^{n} g_i w_i] = g_A \\ dis(A) = v[\left(1 - \sum_{i=1}^{n} w_i\right) [1 - \prod_{i=1}^{n} (1 - f_i)] + \sum_{i=1}^{n} f_i w_i] = f_A \\ uncer(A) = 1 - g_A - f_A \end{cases} \quad (4)$$

where each w_i (i=1,2,…,n) denotes a disjoint contributing weight of the ith sub-claim to support its parent claim. Again, due to space constraints, the formula for the trustworthiness of a claim with an n-claims redundant argument is omitted and can be found in [2].

2.5 Derivation of Disjoint Contributing Weights

Since the first assurance case is acceptable, a certifier usually compares it with some other assurance cases which have the same structure except for leave claims. For instance, in Figure 2, for leaf claim 0G1.7.1, *i.e.*, "Sys R01 traces to HR5 at the CONOPS", another assurance case used by the certifier can have "Sys R01

traces to HR1 at the CONOPS" for leaf claim 0G1.7.1. Then, the evidence node supporting 0G1.7.1 should be "Trace check report for Sys R01". Based on this fact, the framework generates a certain number of assurance cases to infer the disjoint contributing weights for the same structure of two GSNs. This procedure can be illustrated by the following three aspects.

Generation of a set of learning data. The framework first generates a similarity table for each leaf claim in the first input assurance case using GVSM [7]. Specifically, there are totally 5 high-level requirements for 0G1.7.1 in Figure 2 and so the table includes 5 values for *dec*. Each value is calculated by GVSM using Sys R01 as query and high-level requirements such as HR1 as a document. Next, the framework sorts the similarity value in a descending order, and finally chooses the best, average, and worst similarity values as a candidate set to generate a new assurance case. Next, for each leaf claim, the framework randomly chooses a value from the candidate set and an assurance case is generated once all leaf claims are considered. The reason to do so is to make sure no extreme learning data can be generated. For the coupled tanks system, we generate 300 assurance cases.

Deriving values for disjoint contributing weights. For the coupled tanks system, we set the type of argument to be dependent argument as well as the discounting value to be 0.1 for all arguments to denote the uncertainty about the appropriateness of all arguments based on the way we apply a safety pattern to the system. After that, the framework starts the confidence calculation with leaf claims via the D-S theory algorithm. Next, the framework employs the following schema for each argument. For instance, for the top argument, there are three sub-claims 0G1.2.1, 0G1.2.2, and 0G1.2.3 to support the claim 0G1.1.1 so three disjoint contributing weights are defined as w_1, w_2, and w_3 respectively. To find the best configuration of these disjoint contributing weights, the framework considers the following formula (5) to find the value for w_1, w_2, and w_3.

$$\sum_{i=1}^{3} w_i = 0.9, where\ w_i \in \{0.1, 0.2, \dots, 0.7\} \quad (5)$$

Note that the trustworthiness calculation of our framework uses the bottom up strategy, starting with leaf claims. So disjoint contributing weights for all arguments can be derived and the framework finds a set of values for all disjoint contributing weights to assign the first assurance case with the highest *dec* value among the set of assurance cases used as learning data. But if the two input assurance cases have different structures such as the bottom portions in Figure 1, the framework assigns equal weights to all disjoint contributing weights for an argument instead using formula (5).

Apply to the second assurance case. The framework applies the derived values of disjoint contributing weights to the calculation model of the second assurance case whose confidence can be thereby automatically generated instead of manually done by a certifier. The framework accepts the second assurance case generated by the system artifacts [8] via returning 0.72 as the value of variable *dec* as well as 0.77 as the value of variable *conf*. The result matches our evaluation result and expectation.

3 RELATED WORK

Evaluation of assurance cases is not new to the safety critical sectors. Various uncertainty theories or models have been applied to deduce confidence of an assurance case. Most approaches tailor a theory or model to an assurance case so a confidence value can be calculated. For instance, Wang *et a*l. used the Dempster Shafer (D-S) theory as their main calculation model in order to deduce the confidence of an assurance case [2]. Denney *et al*. applied a Bayesian paradigm for uncertainty modelling and assessment [9]. Duan *et al*. considered the application of the Beta distribution as Baconian Probabilities [10]. However, all the current approaches require human input to configure assessment parameters such as how a child node contributes to the belief of its parent node.

4 CONCLUSIONS

While software certification requires human judgement, our preliminary results show that software certification can be leveraged by means of automation when the D-S model is used to simulate a certifier's domain model and experience. Investigations are proceeding into the accuracy of the confidence we are able to achieve within the framework. Lastly, investigations are looking into how other mathematical models might affect the confidence calculation as well as comparison amongst these alternative models.

REFERENCES

[1] EUROCONTROL—European Organisation for the Safety of Air Navigation, "Preliminary Safety Case for ADS-B Airport Surface Surveillance Application, V 1.2," [Online]. Available: https://www.eurocontrol.int/sites/default/files/publication/files/surveillance-cascade-preliminary-safety-case-for-airports-surface-surveillance-applications-201111.pdf.

[2] R. Wang, J. Guiochet and G. Motet, "Confidence Assessment Framework for Safety Arguments," in *Proc. of SafeComp'17*, Trento, Italy, 2017.

[3] E. W. Denney and G. J. Pai, "Safety Case Patterns: Theory and Applications," NASA/TM–2015–218492, 2015.

[4] R. Hawkins, I. Habli, D. Kolovos, R. Paige and T. Kelly, "Weaving an Assurance Case from Design: A Model-Based Approach," in *Proc. of HASE'15*, Daytona Beach, FL, 2015.

[5] Goal Structuring Notation Working Group, "GSN Community Standard Version 1," 2011.

[6] K. H. Gross, A. W. Fifarek and J. A. Hoffman, "Incremental Formal Methods Based Design Approach Demonstrated on a Coupled Tanks Control System," in *Proceedings of HASE'16*, Orlando, FL,, 2016.

[7] G. Tsatsaronis and V. Panagiotopoulou, "A Generalized Vector Space Model for Text Retrieval Based on Semantic Relatedness," in *Proc. of EACL'09*, 2009.

[8] AFRL-VVCAS, "TwoTanksExample," October 2015. [Online]. Available: https://github.com/AFRL-VVCAS/TwoTanksExample.

[9] E. Denney, P. Ganesh and H. Ibrahim, "Towards Measurement of Confidence in Safety Cases," in *Proceedings of ESEM'11*, Banff, Alberta, Canada, 2011.

[10] L. Duan, S. Rayadurgam, M. Heimdahl, O. Sokolsky and I. Lee, "Representation of Confidence in Assurance Cases Using the Beta Distribution," in *Proceedings of HASE'16*, Orlando, FL,, Jan, 2016.