# Eliminative Induction: A Basis for Arguing System Confidence

John B. Goodenough, Charles B. Weinstock, Ari Z. Klein
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA, USA
jbg|weinstock|azklein@sei.cmu.edu

***Abstract*—Assurance cases provide a structured method of explaining why a system has some desired property, e.g., that the system is safe. But there is no agreed approach for explaining what degree of confidence one should have in the conclusions of such a case. In this paper, we use the principle of eliminative induction to provide a justified basis for assessing how much confidence one should have in an assurance case argument.**

***Index Terms*—assurance case, eliminative induction, defeasible reasoning, safety case**

## I. Introduction

At many stages in the design, development, and commissioning of a software-reliant system, claims are made about system behavior and evidence is gathered with the intention of gaining confidence in the truth of these claims. The designer, implementer, and release authorities all want to develop justified confidence that the system will meet user needs safely, securely, reliably, and with acceptable performance. But what evidence leads to an increase in confidence that such properties hold? Why does this evidence do so? What is the justification for saying confidence is increased or is sufficient? In fact, what do we mean by "confidence"?

Our research [1] has been aimed at answering these kinds of questions. We use the notion of *eliminative induction,* first put forward by Francis Bacon and more recently as expounded by L. Jonathan Cohen [2] and David Schum [3], as the basis for a theory of confidence applicable to software-reliant systems. In eliminative induction, a claim is justified only to the extent that reasons for doubting its truth have been eliminated. As reasons for doubt are eliminated (through evidence or argument), confidence in the truth of the claim increases.

Confidence, in this view, is just a function of how many reasons for doubt have been identified and removed. If $n$ reasons for doubting a claim have been identified and $i$ of these are eliminated by argument or evidence, then confidence in the claim is expressed as the *Baconian probability,* $i|n$. ($i|n$ is read as "$i$ out of $n$"; it is not a fraction.)

Having "no confidence" in a claim means no reasons for doubt have been eliminated ($0|n$). Having total confidence means all identified doubts have been eliminated ($n|n$). This notion of confidence has little or nothing to do with the (Pascalian) probability that the claim is true. For example, in the Pascalian approach, probability zero means the claim is never true, and probability one means that it is always true.

In addition to basing our approach on the Baconian notion of eliminative induction, we make use of *defeasible reasoning* concepts (see section II) to generate reasons for doubting the truth of a claim. Potential reasons for doubt are called *defeaters*.

In a previous report [4], we created a notional assurance case [5] for showing that parameters to a medical device will be entered accurately. The case considered how to mitigate various potential hazards leading to inaccurate or unintended manual entry of parameters. Among the hazards considered was the possibility that keypad markings were ambiguous. An assurance case arguing that this hazard is mitigated is shown in Fig. 1,[1] but the assurance case structure does not make explicit the reasons for doubt that are being eliminated by the claims and evidence. Nor does it capture the reasons why the argument was developed as shown and where there might be weaknesses.

A *confidence map* is a structure that explicitly shows the reasons for doubt relevant to a particular argument. A confidence map consists of claims, defeaters, inference rules , and evidence (data). A confidence map is useful both in developing an argument and in evaluating an argument's strengths and weaknesses. The confidence map structure is grounded in the Baconian philosophy of eliminative induction and the use of defeasible reasoning to generate reasons for doubt. This paper explains the potential benefits of a Baconian approach to arguing confidence in system properties, or more generally, the amount of confidence one is justified in having in an assurance case presented in support of some claim. This paper introduces the notion of a confidence map and explains how it can be used.

## II. Defeasible Reasoning

In practice, arguments about system properties are *defeasible*, i.e., the conclusions are subject to revision based on additional information [6]. In the defeasible reasoning literature [7] [8], the ways of attacking an argument are called defeaters.

[1] The notation is Goal Structuring Notation [9] in which claims are in rectangles, evidence is shown in circles, and argumentation strategy in parallelograms. A claim with a diamond at the bottom is to be developed further. The arrows from claim to claim or to evidence can be read as "because".

978-1-4673-3076-3/13 © 2013 IEEE



ICSE 2013, San Francisco, CA, USA
New Ideas and Emerging Results

# Report Documentation Page

*Form Approved*
*OMB No. 0704-0188*



| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED |
|---|---|---|
| **MAY 2013** | | **00-00-2013 to 00-00-2013** |

| 4. TITLE AND SUBTITLE | 5a. CONTRACT NUMBER |
|---|---|
| **Eliminative Induction: A Basis for Arguing System Confidence** | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University,Software Engineering Institute,Pittsburgh,PA,15213** | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S) |
| | 11. SPONSOR/MONITOR'S REPORT NUMBER(S) |

12. DISTRIBUTION/AVAILABILITY STATEMENT
**Approved for public release; distribution unlimited**

13. SUPPLEMENTARY NOTES
**35th International Conference on Software Engineering (ICSE), 18-16 May 2013, San Francisco, CA.**

14. ABSTRACT
**Assurance cases provide a structured method of explaining why a system has some desired property, e.g., that the system is safe. But there is no agreed approach for explaining what degree of confidence one should have in the conclusions of such a case. In this paper, we use the principle of eliminative induction to provide a justified basis for assessing how much confidence one should have in an assurance case argument.**

15. SUBJECT TERMS

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | |
| **unclassified** | **unclassified** | **unclassified** | **Same as Report (SAR)** | **4** | |

There are only three types of defeaters: rebutting, undermining, and undercutting. A *rebutting* defeater provides a counter-example to a claim. An *undermining* defeater raises doubts about the validity of evidence. An *undercutting* defeater specifies circumstances under which a conclusion is in doubt when the premises of an inference rule are true.

As a simple example, we might argue "Tweety can fly because Tweety is a bird." This argument is supported by the inference rule, "If X is a bird, X can fly." But suppose we notice that Tweety is actually a penguin (a rebutting defeater), or that Tweety is actually a bat (an undermining defeater attacking the premise that Tweety is a bird), or that although Tweety is a member of a bird species that flies, Tweety is a juvenile (an undercutting defeater because the premise is true but the conclusion is uncertain). In each of these cases, we have raised doubts about the validity of the argument and the correctness of its conclusion. Identifying defeaters and removing them (by showing that they are false) is the essence of our approach to building and assessing confidence in an assurance case argument.

## III. Building Confidence by Doubt Refinement

To build confidence in the claims supported by an assurance case, we seek to understand the reasons for doubt implicit in the case, i.e., we look for defeaters associated with the various elements of the case:

- For each claim, we look for counter-examples (rebutting defeaters).
- For each piece of evidence, we look for reasons the evidence might not be valid (undermining defeaters).
- For each inference rule, we look for conditions under which the rule would not hold (undercutting defeaters).

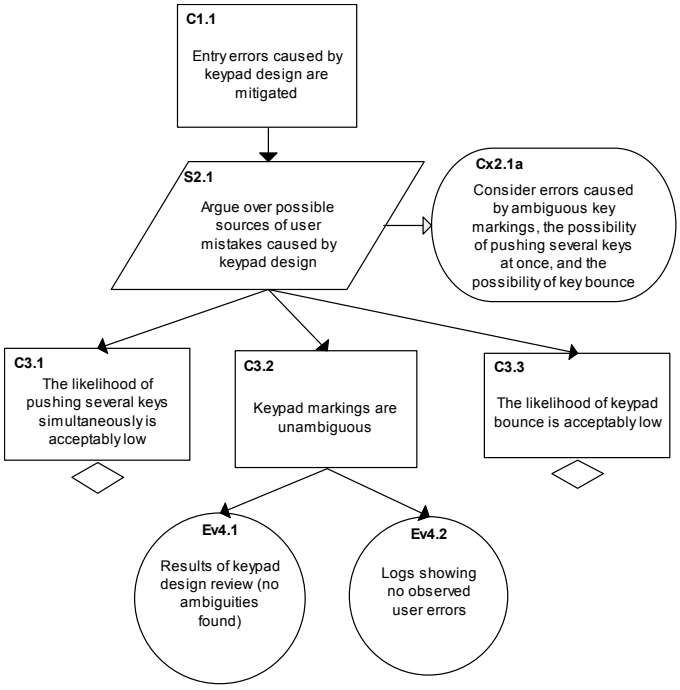


Fig. 1. Case for entry error mitigation.

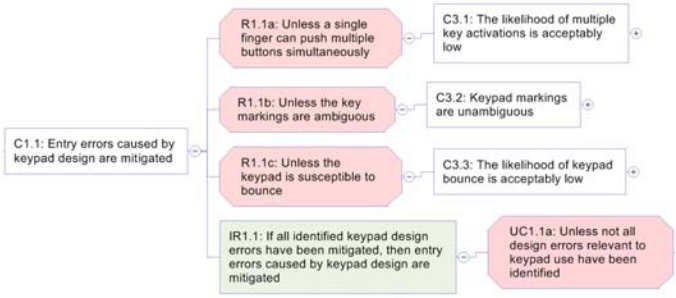


Fig. 2. Top-level defeaters.

In applying this approach to the case shown in Fig. 1, we first consider the claim "Entry errors caused by keypad design are mitigated" and ask what would cause us to doubt the truth of the claim. Obviously, we need to look for possible keypad design errors that could cause entry mistakes. In this example, we suggest three design errors as rebutting defeaters and state how these errors would be recognized:

- An attempt to push a key enters two values because two keys are pushed unintentionally,
- The keypad markings are interpreted differently by different users.
- A single key push causes two inputs to be received (key "bounce").

In order to eliminate the defeaters, the rest of the case must show that,

- The likelihood of pushing two keys instead of one is acceptably low.
- Keypad markings are unambiguous.
- The likelihood of keypad bounce is acceptably low.

The associated confidence map is shown in Fig. 2. Claims are presented in clear rectangles (labeled with a "C"), inference rules in green-shaded rectangles (labeled "IR"), and defeaters in red-shaded octagons labeled with an "R", "UC" or "UM" (for rebutting, undercutting, and undermining defeaters, respectively). For readability, each statement of a rebutting or undercutting defeater begins with the word "Unless". Each statement of an undermining defeater (such as those shown later in Fig. 3) starts with the word "But".

The confidence map makes explicit the inference rule that is implicit in Fig. 1 in connecting claims C3.1, C3.2, and C3.3 to claim C1.1. The implicit rule is "If all identified keypad design errors have been mitigated, then entry errors caused by keypad design are mitigated." This rule is, of course, defeasible—if a credible keypad design error has *not* in fact been identified, it is possible for the premise to be true (all identified keypad design errors mitigated) and the conclusion uncertain. This situation is captured in the undercutting defeater associated with the rule: "Unless not all design errors relevant to keypad use have been identified". The undercutting defeater captures conditions under which the premises of a rule are insufficient to justify its conclusion. In other words, the undercutting defeater provides a reason for doubting that the inference is sound.

At this stage in developing the confidence map, a reviewer can suggest additional rebutting defeaters or clarification of proposed defeaters. We generally find it illuminating to focus

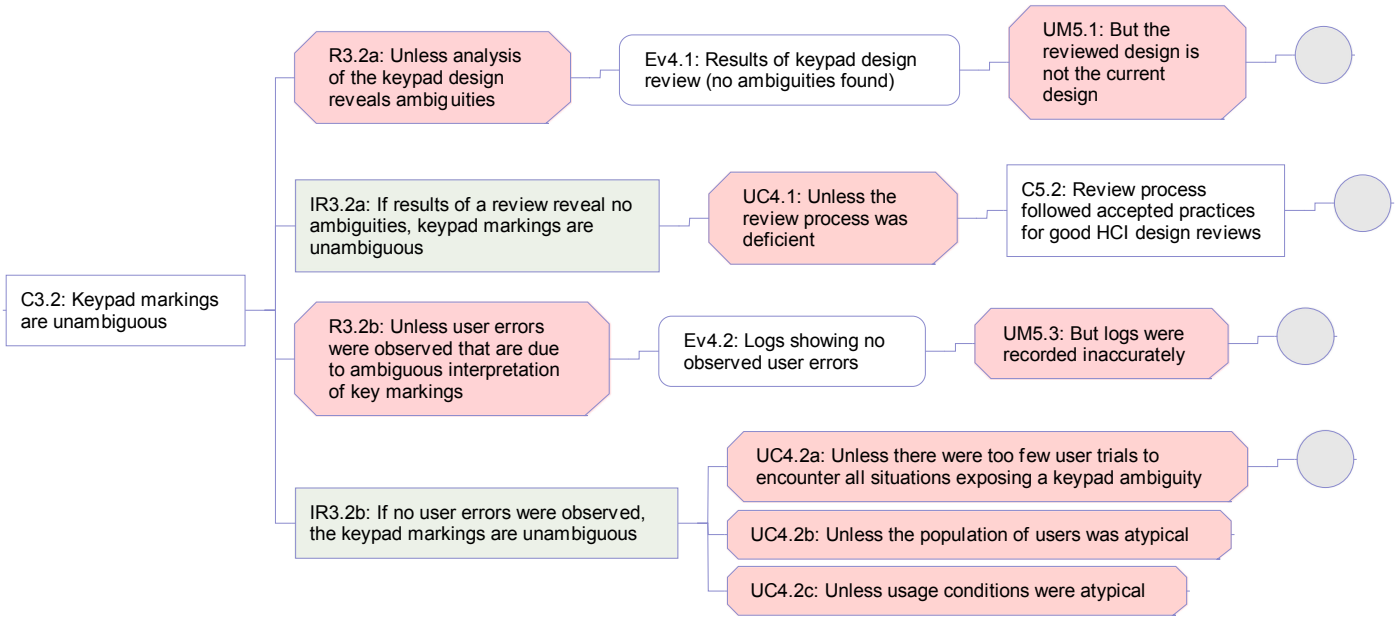


Fig. 3. Refinement of doubt.

on identifying defeaters *before* considering how to develop the argument further by deciding how to eliminate the defeaters. A key contribution of the confidence map is to explain why an argument is being developed in a certain direction, namely, in the direction required to eliminate defeaters. The next level of argumentation depends on the set of defeaters that have been identified.

To illustrate our concepts more completely, we develop the confidence map further by considering defeaters for claim C3.2, "Keypad markings are unambiguous" (see Fig. 3). There are at least two kinds of information that could cause us to doubt the claim:

- A review of the keypad by experts could find specific examples of how the keypad markings could be ambiguous under certain operating conditions (R3.2a).
- During test and evaluation or initial operation, a number of keypad entry errors might be logged. If some of them are attributed to ambiguities caused by the design, the presence of such errors would certainly demonstrate that C3.2 is false (R3.2b).

In this way we refine one reason for doubting claim C1.1, namely, R1.1b, into two more specific reasons for doubt—the defeaters for claim C3.2. This process of doubt refinement continues until we decide there is no further useful insight to be gained.

Continuing our explanation of the confidence map shown in Fig. 3, consider defeater R3.2a, "Unless analysis of the keypad design reveals ambiguities". R3.2a raises the possibility that an expert review of the keypad design would find potential ambiguities in the keypad markings. This defeater can be eliminated directly if the results of such a review say no ambiguities were found (Ev4.1). However, we can raise doubts that make the evidence meaningless. In this case, if the reviewed design is not the current design (UM 5.1), we can't conclude that the design is unambiguous.

Because we are using Ev4.1 as support for C3.2, we need to state explicitly the inference rule that is being used to infer C3.2 from the evidence. IR3.2a explicitly states this connection, namely, "If results of a review reveal no ambiguities, keypad markings are unambiguous." Of course, this inference is defeasible—under some conditions, the lack of discovered ambiguities does not necessarily imply the keypad markings are unambiguous, i.e., under such conditions, the evidence would not be sufficient to support the claim. We capture such conditions in the undercutting defeater, UC4.1, saying in essence that if the review process is deficient, a lack of discovered ambiguities would not give us confidence that the keypad markings are unambiguous. We then go on to suggest that this reason for doubting a lack of ambiguity would be eliminated if we knew that the review process followed accepted practices for good human-computer interaction (HCI) design reviews (C5.2).

Similarly, Ev4.2, "Logs showing no observed user errors", is used to eliminate defeater R3.2b. Although the assurance case in Fig. 1 stops with Ev4.2, the eliminative induction approach to confidence requires that we consider under what conditions this evidence could be irrelevant or insufficient (see UM5.3 and UC4.2a, b, and c).

By this time, it may seem that we can raise doubts *ad infinitum*. In principle this is true, but in practice, doubt refinement stops when we decide that a defeater or claim is sufficiently obvious that no further doubts are credible, i.e., no increase in confidence will result from further development of the argument. We indicate this in a confidence map with a shaded circle. For example, the confidence map in Fig. 3 says that defeater UC4.1 is eliminated because we have no doubt about the validity of claim C5.2. On the other hand, the map says we have not yet addressed undercutting defeaters UC4.2b and c; they remain as uneliminated reasons for doubt. Further

argumentation and evidence is needed to eliminate these defeaters.

## IV. Evaluating Confidence

We started this paper by asserting that confidence is increased as the number of defeaters is eliminated. What is the number of defeaters to be considered in deciding how much confidence we have in the truth of C3.2?

In general, a confidence map structure shows reasons for doubt being refined from general reasons to more specific reasons. So although the confidence map for C3.2 clearly contains eight defeaters, each of the R3.2 defeaters has been replaced with a refined set of defeaters, e.g., R3.2a has been replaced by UM5.1; if this defeater is eliminated, R3.2a is eliminated. In general, the total number of defeaters relevant for evaluating confidence is the number of defeaters closest to or at the leaves of the confidence map. On this basis, the confidence map for C3.2 specifies six defeaters relevant to claim C3.2.

Of these six defeaters, all are considered to be eliminated except for UC4.2b and c, i.e., only one of the three defeaters associated with IR3.2b is eliminated. This is expressed by giving IR3.2b a Baconian probability of 1|3, i.e., we have two unsatisfied reasons for doubting that IR3.2b is false. Unsatisfied reasons for doubt constitute *residual doubt* and are the basis for reduced confidence in the truth of a claim. Similarly, when considering claim C3.2, four out of six reasons for doubting the truth of C3.2 have been eliminated, so our (Baconian) confidence in C3.2 is expressed as 4|6.

The denominator is not so interesting in this expression. We are primarily interested in residual doubt (the number of uneliminated defeaters) because these are the remaining reasons for doubting the truth of a claim. Documenting these remaining reasons for doubt helps to explain why our confidence in the truth of a claim is less than total. We can then decide whether these remaining doubts are important enough to deserve additional effort to remove them.

We are exploring other possible ways to best summarize the information in a confidence map. Styles of argument and their effects on confidence is also a subject of our current research. For example, the argument shown in Fig. 3 is "multi-legged", i.e., an argument in which a claim is supported by independent evidence and inference rules. We are investigating how our approach to confidence explains why multi-legged arguments seem to give us more confidence in the top-level claim.

## V. Summary

This paper presents a new application of eliminative induction and defeasible reasoning concepts as the basis for justifying confidence in claims about system properties. The notion of a confidence map allows reasons for doubt to be documented explicitly in a reviewable form. We are currently working on further development of these notions and their utility. For example, the role of evidence as a basis for increasing confidence is tricky. In reasoning by eliminative induction, confirmatory evidence (e.g., a successful test result) is, by itself, meaningless. In the Baconian view, evidence only gains meaning when we understand what reasons for doubt it eliminates and how confidence in certain claims is increased by such eliminations. The confidence map shows the significance of evidence in increasing confidence in system claims.

The fact that additional doubts can always be identified is disturbing for most people who encounter these ideas for the first time. But the potential for additional doubt is, in practice, the reality. For example, a safety case is always subject to revision when additional information is discovered. Nonetheless, at some point, we decide that a sufficient set of doubts have been identified and resolved, and we move on. Our approach only makes this fact more explicit and reviewable.

In summary, an assurance case provides an argument and supporting evidence as *justification* for a claim. But we seek to provide justification for *belief* in a claim, and we do so by identifying and eliminating defeaters relevant to the claim.

## Acknowledgment

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

This material has been approved for public release and unlimited distribution. DM-0000209.

## References

[1] J. Goodenough, C. Weinstock and A. Klein, "Toward a Theory of Assurance Case Confidence," Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2012.

[2] L. J. Cohen, An Introduction to the Philosophy of Induction and Probability. Oxford: Clarendon, 1989.

[3] D. Schum, The Evidential Foundations of Probabilistic Reasoning. Evanston, IL: Northwestern University Press, 2001.

[4] C. B. Weinstock and J. B. Goodenough, "Towards an Assurance Case Practice for Medical Devices," Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2009.

[5] ISO/IEC, "15026-2:2011 Systems and software engineering -- Systems and software assurance -- Part 2: Assurance case," 2011.

[6] Metaphysics Research Laboratory, Center for the Study of Language and Information, "Stanford Encyclopedia of Philosophy: Defeasible Reasoning," 2009. [Online]. Available: http://plato.stanford.edu/entries/reasoning-defeasible.

[7] J. Pollock, "Defeasible Reasoning," in Reasoning: Studies of Human Inference and Its Foundations, J. E. Adler and L. J. Rips, Eds., Cambridge University Press, 2008, pp. 451-469.

[8] H. Prakken, "An Abstract Framework for Argumentation with Structured Arguments," London: Argument & Computation, vol. I, no. 2, pp. 93-124, 2010.

[9] Goal Structuring Notation Community Standard, Version 1. York, UK: Origins Consulting Limited, 2011. [Online] Available: http://www.goalstructuringnotation.info/documents