The Increasing Risks of Risk Assessment: On the Rise of Artificial Intelligence and Non-Determinism in Safety-Critical Systems

C.W. Johnson[1]

School of Computing Science, University of Glasgow

Glasgow, U.K.

[If you want to add a full address, email, etc, use a footnote]

Abstract *Risk assessment plays a key role in Safety Management Systems. For more than forty years, likelihood and consequence have been used to guide the allocation of finite resources. Standards, such as IEC61508 and the DO-178 series, extended these concepts to support the development of software related systems. Human reliability analysis developed risk assessment techniques to represent and reason about operator 'error' and management failure. However, new challenges raise questions about the utility of traditional approaches to the development of safety-critical systems. The introduction of artificial intelligence within autonomous systems makes it hard to reason about the probability and consequences of adverse events when control applications must use previous training sets to guide their response to novel situations. This paper struggles to retain the foundations of risk assessment as a tool for safety engineering in the face of these new challenges for the development of safety-critical applications.*

1 Introduction

Risk assessment guides the development of safety-critical systems. The consequence and likelihood for a range of hazards can be combined to focus the allocation of finite development resources. Engineers and managers coordinate their activities to mitigate the impact and reduce the likelihood of adverse events (Johnson, 2010). A host of qualitative and quantitative approaches support the use of risk assessment ranging from Fault Trees to Markov models and Common Cause Analysis (Ostrom and Wilhelmsen, 2012). Risk matrices map out different levels of risk within the cells that denote particular combinations of likelihood and consequence. The impact of mitigations can then be shown as movement between these columns and rows (Rausand, 2013). At the same time, hazard analysis techniques including both HAZOPS and FMECA provide structured tools for

[1] johnson@dcs.gla.ac.uk

teams of designers to represent and reason about the precursors to adverse events (Ericson, 2016). These, in turn, can be included as evidence within the leaf nodes of a safety-case to show that an implementation is acceptably safe (Maguire, 2006).

2 Existing Limits on Probabilistic Risk Assessment

Risk assessment techniques have evolved to support the design of software-intensive applications and also of interactive systems; where human factors play a critical role in safe and successful operation. Subsequent sections will argue that similar changes are required if these approaches are to support the future development of complex, safety-critical applications that embed AI and ML algorithms.

2.1 Human Factors

The early applications of risk assessment techniques focussed narrowly on hardware; statistical observation of previous systems yielded valid predictions of future reliability. In particular, the derivation of failure probability distributions to reflect both burn-in and burn-out times enabled engineers to make accurate judgements about the impact on reliability of maintenance and replacement techniques. The use of graphical approaches, including Fault Trees, helped to generalise this Probabilistic Risk Assessment (PRA) from individual component failures to more complex, system hazards; with particular successes in the development of nuclear (US NRC, 1984) and military applications (Saleh and Marais, 2006). A key strength of PRA was the ability to drive predictions of future risk from observations of past performance. Monte Carlo simulations yielded further benefits. The synthesis of more complex systems behaviors could help predict potential traces of interaction based on statistical data about individual sub-systems, even when the eventual application had yet to be developed (Greenland, 2001).

The statistical foundations for PRA created particular problems when it was extended to more interactive applications. In such environments, it proved hard to ignore the impact that human operators might have upon system behaviors. Individual cognitive and perceptual factors had a profound impact on error probability and were difficult to generalize between complex applications.

The desire to include the impact of operator interaction in risk assessments led to the development of Human Reliability Analysis (UK HSE, 2009). The intention was to assess the human contribution to risk using techniques that were integrated with existing methods for PRA. A number of factors complicated this integration.

Limits of Determinism. From the perspective of reliability engineering, non-determinism implies that the sample inputs do not always yield the same outputs. Risk assessment relies on probability distributions to characterize this variance. However, the use of such techniques raises problems when moving from hardware to more interactive applications. In particular, probability distributions for the variance of human performance are determined by a host of individual and contextual factors.

Limits of Induction. Inductive reasoning combines multiple premises, all of which are usually assumed to be true, to support particular conclusions. It is informally known as bottom-up reasoning. Evidence can be obtained for probability distributions through inductive reasoning. This relies on statistical observations of past performance to inform predictions of future behavior. We can subject hardware to many thousands of hours of operation to yield data on the likely frequency of future failure modes. In a similar way, we can employ user studies to determine the likely frequency of human error during the operation of complex safety-related systems. However, this raises numerous concerns. Experimental effects cannot easily be eliminated when, for example, users know that they are interacting with simulated control systems rather than safety-critical processes. It is also difficult to sustain the longitudinal studies needed to provide statistically reliable results, especially when highly skilled operators are often a scarce resource;

Limits of Deduction. Deductive reasoning supports particular conclusions by establishing the truth of multiple premises. This is a 'top-down' approach. Many existing risk assessment techniques, which support hardware analysis, rely on deductive reasoning. Functional and structural decomposition can be used to derive the probability of more complex failures based on the past performance of individual sub-components. For example, the likelihood of a fault tree is calculated from the disjunction and conjunction of basic events; for which we have probabilities. Deductive approaches are harder to apply to human behavior where physiology, perception and cognition remain active areas of research. Without some appreciation of these factors it is hard to determine how an initial mistake might influence the likelihood of subsequent slips or lapses (Reason, 2000).

Limits of Context. A final concern over the introduction of human error and resilience into probabilistic risk assessments is the difficulty in assessing the impact of changing contextual factors. In traditional, hardware focused applications of PRA, it is possibly to control or exclude a host of influences that might otherwise have a profound affect on component reliability – including vibration, heat, pressure etc. This is far harder when humans play a key role in the operation of complex systems. While some factors such as alcohol and drug misuse can be controlled, it is typically impossible to entirely exclude their potential influence on behavior. Similarly, probability distributions that might characterize the impact of

organizational and social factors, of fatigue and of stress are all hard to assess and to validate.

Early approaches to Human Reliability Analysis addressed these challenges by exploiting relatively simple deductive techniques. They decomposed higher-level activities into their component sub-tasks. This mirrored the manner in which existing risk analysis techniques used functional or structural decomposition to identify the hazards associated with hardware components. However, the objections, listed above, were addressed through the inclusion of modifying factors, including the impact of time pressure, equipment design, fatigue etc, were then applied to the base sub-tasks to derive estimates of Human Error Potential.

Early forms of Human Reliability Analysis, including THERP (Swain and Guttman, 1983) and HEART (Williams, 1985), failed to consider the impact of working environment and organizational context. Critics also claimed that they were 'psychological vacuous'; they lacked any explicit basis in cognitive psychology (Hollnagel, 1998). A subsequent generation of HRA techniques used Performance Shaping Factors to modify the raw probability that operators might fail to perform necessary actions or might intervene when not expected. The effects of PSFs were determined with reference to underling cognitive and perceptual models of performance, including ATHEANA (US NRC, 1996) and CREAM (Hollnagel, 1998).

2.1 Software

The identification of weaknesses in existing risk assessment techniques and the development of potential solutions do not always lead to the uptake of more advanced methods. In particular, the UK Health and Safety Executive's (2009) review of HRA argues that most human factors risk assessments are narrowly based on expert judgment rather than the statistical foundations of hardware analysis or the cognitive approaches proposed in CREAM or ATHENA. In contrast, a number of techniques have extended the application of risk assessment to guide the development of software intensive safety-critical systems.

The early applications of probabilistic risk assessment did not have to consider the integration of software because most safety-related applications relied entirely on hardware devices until the late 1980s. However, the increasing pressure to enable rapid reconfiguration and the desire to exploit a growing array of Commercial Off the Shelf (COTS) microprocessors meant that new techniques needed to be developed (Butler and Finelli, 1993).

Limits of Determinism. Software is deterministic in the sense that the same inputs to the same code should always yield the same results. In practice, howev-

er, things are seldom this simple. The impact of any sequence of operator commands and sensor readings on the behavior of code depends upon the underlying state of the system. It is increasingly difficult to determine both the state of the system and the impact of particular command sequences without significant forensic effort given the complexity of modern microcode and the range of software architectures embedded in processors/'Smart' sensor-actuators. At a higher level, operators and managers are often surprised by the consequences of interaction when they are unaware of the detailed configuration of their systems. It is increasingly hard to enumerate all of the potential interactions that can arise between dozens of highly networked systems. We can increase determinism by placing limits on software. For example, the time triggered network protocols used in many recent modular avionics systems allocate particular time slots to particular processes. This supports predictions about network traffic and the consequent behavior of any safety-critical software that relies on this communication. Similarly, the difficulty of predicting how long it will take to retrieve variables into a CPU can be eased by disabling any associated caches; supporting predictions about Worst Case Execution Time. These controls help to increase the apparent determinism of complex, safety critical software. They support predictions about the future behaviour of complex systems.

Limits of Induction. As with the integration of human factors into probabilistic risk assessment, software also creates challenges to the use of inductive techniques based on statistical observations of previous failures. These include the exponential complexity of computer algorithms. Any code with a sequence of n decisions has up to 2^{n} paths. There is no general method for identifying infeasible paths; such a method would imply a solution to the halting problem. Path complexity undermines inductive approaches that can be used to assess the reliability of software related systems. Code coverage metrics, including MCDC, measure the degree to which the source code of a program is executed when a particular test suite runs. However, we cannot estimate the probability that code contains a bug by exhaustive testing; following Dijkstra's maxim that 'testing proves the presence of bugs, not the absence'. Even if a sustained verification discovered few additional errors, different testers using different techniques might uncover many more bugs. Fault injection provides a proxy; developers deliberately insert a known number of bugs into the code. Independent testing teams work until they find all of the deliberately inserted errors, hopefully finding many other bugs. The proportion of injected errors found versus the total number introduced can be related to the likelihood of any remaining bugs. However, it makes no sense to halt the verification process until all the injected faults have been removed. At the end of all this, even if every deliberate fault is discovered, we have no means of assessing the residual probability of software failure. Alternative approaches have been developed, for example trying to compute the raw probability that code contains a bug based on statistical methods. However, these approaches hit against problems that include the expansion ration between a high-level language and the underlying machine code delivered from a compiler.

Measurement of software failure rates must include the speed of the processor and the locality of reference within any particular trace (Miller et al, 1992). In consequence, they have largely been abandoned.

Limits of Deduction. Given the inherent difficulties in the use of inductive techniques to quantify the future probability of software failures, deductive techniques offer significant benefits; especially where formal mathematical approaches can be used to represent and reason about potential errors. Deductive verification proceeds by developing a specification of the system under consideration together with a number of proof obligations that are intended to demonstrate the correctness of the underlying design. Examples of this approach include the application of PVS and HOL. Alternatively, model based techniques proceed by checking whether particular theorems, typically expressed in Linear Temporal Logic or Computational Tree Logic, hold across all given states of a potential design. This can be seen as a hybrid of induction, to generate the states of the system, and deduction, to determine whether or not a theorem holds true. In both cases, there are theoretical and practical concerns when using these techniques to drive probabilistic risk assessment of safety-related software. Model checking can only be applied to a subset of systems with potentially infinite states. It can also be hard to ensure that an eventual implementation accurately satisfies the assumptions that guided the mathematical modeling. Legacy systems and intellectual property barriers prevent the use of formal modeling to accurately represent and reason about many of the components in complex systems. Although the use of deductive, formal reasoning increase confidence in the correctness of software applications, we cannot assume that it reduces the probability of failure to zero nor does there seem to be any agreement on the impact of these methods on residual risk.

Limits of Context. A final barrier to the estimation of software failure probabilities arises from the differences between test environments and the eventual context in which software is deployed. Software offers significant benefits in terms of flexible reconfiguration and consequent reuse. The parameters that guide a particular application can be tailored to support different environments. Equally assumptions that were valid in previous contexts can led to software related failures in new situations. This is exacerbated because many bugs stem not from implementation errors but from flaws in the underlying requirements. This also affects hardware but such requirements failures are, typically, compounded when software is specifically intended to support rapid reconfiguration. Legacy code again creates problems when operators do not have access to the source or to the underlying assumptions that guided initial development processes. In such situations, it may only be feasible to use black box verification techniques that suffer from the limitations of inductive testing, mentioned above. It is hard to underestimate the consequences of these barriers given that the widespread adoption of risk assessment has been mirrored by the increasing integration of software into safety-related applications. One solution is to move away from objective forms

of software failure probabilities and instead to rely on expert judgment. However, this is as controversial, and arguably as (un)reliable, as subjective predictions of human error (Authen and Holmberg, 2012).

A number of development standards have been proposed in the absence of effective means for estimating the likelihood of software failure in safety-critical systems. These include IEC 61508 for programmable devices, IEC 62279 for rail, ISO26262 for automotive software and EUROCAE ED-109A/ED-153 for ground-based software in aviation. Although there are significant differences, these documents build on common foundations in traditional risk assessment. They minimize any reliance on software failure rates by focusing instead on the likelihood and consequence of hazards associated with Equipment (and processes) Under Control (EUC). These are distinguished from the safety functions that are intended to reduce any residual risks to an acceptable level. The greater the level of risk associated with the EUC then the greater will be the required level of risk reduction and hence the desired level of integrity that must be met by any software related safety function. In 61508, this is encapsulated within the notion of a Safety Integrity Level (SIL). At SIL4, strong requirements are placed on the software development techniques intended to meet the target failure rates identified in the standard. Defensive programming is highly recommended; including the use of error detection and correction codes. Diversity may also be required when the same bug in any common code will undermine redundant or fallback processes. In other words, the development techniques associated with particular SILs are intended to achieve levels of reliability in the safety processes that protect EUC, without having to directly measure the actual failure rates associated with the software that supports those processes. The success of these standards can be measured by their widespread adoption. The identification of SILs as a proxy for the calculation of software failure rates provides means of retaining risk assessment as the foundation for system safety.

IEC 61508 and similar standards do not resolve the problems associated with human reliability assessment (HSE, 2001). It is also hard to establish any statistical evidence to quantify the benefits of standards like 61508 (HSE 1995, Foord et al, 2011). Some studies have examined the relationship between particular software development techniques and our ability to meet target levels of reliability demanded by particular SILs (Littlewood and Strigini, 2013).

3 Machine Learning and Emerging Risks for Risk Assessment

We have identified continuing concerns over the integration of human factors and software failures into the qualitative and quantitative risk assessments that underpin safety-critical systems development. These stem from the limits of deductive and inductive reasoning; compounded by the impact of non-determinism and by

differences between the intended and actual context of use in critical infrastructures. The following paragraphs build on this analysis and identify a new generation of challenges that share common roots with those identified in previous sections.

Standards such as IEC61508 and ISO26262 sidestep the need to directly assess the likelihood and consequence of software failure. SILs provide a proxy where development resources are focused in proportion to the target level of reliability required to protect end-users and the public from the risks associated with equipment under control. At higher levels of integrity, more rigorous requirements are placed upon the recommended software development practices. Conversely, other approaches are explicitly 'not recommended' because they are perceived to be incompatible with the necessary levels of reliability. These 'not recommended' approaches include machine learning. This is justified because noisy data, non-convex objectives, model misspecification, and numerical instability can all cause undesired behaviors in machine learning systems. Detecting actual implementation errors can be extremely difficult (Selam et al, 2017). However, such prohibitions are increasingly hard to sustain when more and more industries are seeking to integrate the benefits of ML and AI into safety related applications.

Limits of Determinism. Machine learning exploits various forms of induction to generalize models of decision-making based on a number of training sets. A key attribute is that the associated algorithms can automatically identify salient features of the training set without being explicitly guided to extract them. These features are then recognized and used to guide future decision-making when previously unseen cases are presented. Machine learning algorithms include artificial neural networks, regression analysis, Bayesian techniques etc. These approaches pose particular concerns for the development of safety-critical systems. They can increase apparent non-determinism. ML algorithms often seem to exhibit unexpected behaviors when developers lack a detailed understanding of the inductive processes that guide future decision-making. At the heart of this concern is not only the possibility that a machine learning algorithm will fail to recognize and respond to potential dangers but that developers may not be able to predict when it will fail in this manner.

The prohibition on machine learning at higher levels of integrity in existing software safety standards cannot be sustained in the face of innovative applications for these technologies. New generations of airborne and ground-based vehicles rely on ML algorithms for collision avoidance but also to optimize on-board systems; in particular to sustain battery life and support senor fusion. It is hard to envisage how these platforms might be developed without the use of machine learning.

The incentives for innovation outweigh the arguably conservative restrictions embodied within some existing safety standards. Regulatory agencies are under

increasing pressure to facilitate the deployment of these applications even though they use software techniques that cannot meet existing reliability requirements (Johnson, 2016). In 2012, the United States Congress mandated the FAA to issue specific guidance on the regulations governing the introduction of Remotely Piloted Airborne Systems into the national airspace. The intention was to provide a comprehensive framework that would replace the ad hoc exemptions that supported the commercial operation of these drones. However, the anticipated deadlines were hard to meet and by 2015 more than 1,000 companies had been issued FAA333 exemptions. This caution is justified –the use of waivers enables the FAA to gain sufficient operational expertise to inform the subsequent development of regulatory requirements. The UK government has advocated a Code of Practice to promote safety. Such guidance can be extended to support the integration of ML algorithms into safety-related applications, for instance by constraining the use of reinforcement techniques. Reinforcement increases non-determinism when dynamic programming alters underlying algorithms in response to interaction with an eventual operating environment. The key concern is that a system, which initially met an acceptable level of safety, might learn unsafe responses

Limits of Induction. Approval for the integration of machine learning in safety-related applications depends upon our confidence that it will recognize and respond to potential hazards. The proponents of these technologies almost universally rely on longitudinal field trials to convince politicians and the public that they can achieve acceptable levels of safety. They reject the use of proxies for software failure rates. Rather than assess SILs as a guide for the allocation of finite development resources, many of the proponents of machine learning justify their systems by demonstrating equivalence with conventional applications. In other words, the application of machine learning is acceptably safe if the overall system is at least as safe as an application that does not use these technologies. Waymo (formerly Google Cars) has driven almost two million miles in autonomous mode. These verification techniques sustain inductive safety arguments that previous sections of this paper have criticized as a basis for software safety. How can we be confident that longitudinal testing has mitigated software failures to an 'acceptable' level when we cannot test all potential sequences of interaction in uncertain and changing contexts of operation? When do we know that we have conducted sufficient tests to deploy the technology into a particular environment? Can we ensure that controlled tests accurate resemble the context of use or if we test in the eventual working environment how can we be sure that the tests do not create unacceptable risk for the general pubic? How do we ensure that our tests have considered a sufficient range of interactions between ML algorithms and human end-users (Weiner and Smith, 2016)?

In the USA, individual state laws vary significantly and "no state has fully determined how existing traffic laws should apply to automated vehicles" (UK DfT, 2015). Four states have explicit provisions supporting the introduction of this technology. Fifteen have rejected bills related to automated driving even though

the US Department of Transport and the National Highway Traffic Safety Administration (NHTSA) remain committed to the introduction of these technologies. Similar inconsistency can also be seen within Europe. For example, the Germany Federal Highway Research Institute has argued that fully automated vehicles do not comply with existing traffic law. Each Federal state can grant exemptions from the German Road Traffic Licensing Regulations to allow the longitudinal tests required for inductive safety arguments 'provided there is a driver in the driver's seat who has full legal responsibility for the safe operation of the vehicle' (UK DfT, 2015). In France, specific zones have been established for testing, including changes to driver training. There is also provision to allow 'large-scale' testing of self-driving cars and trucks. Sweden has followed a similar approach, allowing tests as part of the Volvo 'Drive Me' project in restricted areas around Gothenburg.

Limits of Context. One of the key objectives in the empirical analysis of AI and ML algorithms is to determine whether the initial training sets enabled the support sufficient generalization for the platform to safely respond to the broad range of environmental conditions that might be anticipated through eventual deployment. This represents an evolution of the previous generation of "proven in use" arguments that have been challenged as a basis for the verification of safety-critical software. IEC 61508-7 Annex D explains how these arguments may be used when, for instance, developers do not have access to underlying source code. It is expected that:

- the previous context of operation was the same or sufficiently close to that in which the new system will be applied;

- if the context differs in any way then analytical techniques and testing must demonstrate that the likelihood of unrevealed faults is low enough to achieve the required SIL for the safety function using that code.

EN 50129 also provides a basis for proven in use arguments. One million hours operation time and at least 2 years experience with different equipment including safety analysis, detailed documentation also of minor changes during operation time is recommended for SILs 3 and 4. However, none of these standards explains the basis on which these 'proven in use' arguments might be extended to support the development of appropriate training sets to demonstrate the acceptability of AI/ML in safety related systems. Salay et al (2017) summarize the implications of this approach for the ISO26262 automotive software safety standard. They argue that there is an assumption in this and other standards that that component behavior is fully specified and each refinement can be verified with respect to its specification. However, this cannot be sustained in AI/ML applications where a training set is used in place of a specification and the training set by definition cannot fully determine all potential operating environments. "Furthermore,

the training process is not a verification process since the trained model will be 'correct by construction' with respect to the training set, up to the limits of the model and the learning algorithm".

From a practical point of view, companies across many different industries have responded in one of two ways – through the development of aggressive training environments and through the control of eventual operating environments. The first approach demonstrates acceptable levels of safety through the development of training sets and then the use of sustained testing environments that show the platform is capable not simply of meeting anticipated operating conditions but also of responding safely in extreme conditions. This may include 'adversarial AI' where it is assumed that external agents deliberately seek to undermine particular platforms; for instance through LIDAR hacking to disrupt or direct the operation of autonomous vehicles.

The use of training sets and test conditions that place the system in 'extreme' operating environments may increase confidence that AI/ML algorithms are capable of coping safely with eventual demands in operation. At the same time, the application of these algorithms is further supported by techniques that constrain or control the working environment. For example, by establishing segregated airspace for Remotely Piloted Airborne Systems or using separate traffic lanes for autonomous vehicles. These constraints help to ensure that safety-related applications meet situations that have previously been encountered during the training of the associated algorithms. As we have seen, however, there are strong commercial reasons to challenge this segregation. It can also be hard to ensure the integrity of constraints on the context of operation – for instance when conventional emergency flights need to cross the segregated airspace devoted to RPAS operations.

There is a clear role for conventional risk assessment techniques to identify the range of scenarios that might be used during the more aggressive forms of training/testing mentioned above. They can also be applied to characterize the constraints that should be placed on eventual operating environments – for example, to consider the probability and impact of those constraints being violated. However, such approaches need to be reinforced by deductive arguments to demonstrate the correctness of the AI/ML algorithms that are embedded within safety-related applications.

Limits of Deduction. A key reason why the integration of AI and ML algorithms poses such a threat to the use of risk assessment is that there are few recognized means of deductive reasoning that might be used to inform the use of inductive empirical approaches. In other words, it is hard to identify similar mechanisms to the functional and structural decomposition that has supported the use of Fault Trees and Cause Consequence Diagrams in existing approaches. Initial steps have been proposed by Ramanathan et al (2016); ``While we depend on the

flawless functioning of such intelligent systems, and often take their behavioral correctness and safety for granted, it is notoriously difficult to generate test cases that expose subtle errors in the implementations of machine learning algorithms… the validation of intelligent systems is usually achieved by studying their behavior on representative data sets, using methods such as cross-validation and bootstrapping". In contrast, they propose the use of symbolic decision procedures coupled with statistical hypothesis testing to validate machine-learning algorithms. They have applied this approach to identify bugs, including bit flips, in implementations of the k-means algorithm that could not easily have been identified using standard validation methods including randomized testing. However, their approach is far from the panacea to the integration of AI into safety-related applications. It remains laborious and time-consuming especially when these approaches are integrated into complex cyber-physical systems.

The need to develop forms of deductive reasoning to improve the correctness of ML algorithms forms part of a wider argument advanced by leading researchers in Machine Learning that 'Advances in Artificial Intelligence Require Progress Across all of Computer Science': "Although AI will be an engine for progress in many areas, creating real-world systems that realize these innovations will in fact require significant advances in virtually all areas of computing, including areas that are not traditionally recognized as being important to AI research and development… future AI systems will not only draw from methods, tools, and themes in other areas of computer science research, but will also provide new directions for research in areas such as efficiency, trustworthiness, transparency, reliability, and security" (Hager, Bryant, Horvitz, Matarić, and Honavar 2017). These authors identify the development of formal methods as a key enabler for the deployment of AI techniques in dependable applications. However, despite a number of recent research initiatives there are no formal standards or widely accepted frameworks for constructing, manipulating or reasoning about the neural networks and other ML approaches being embedded within a host of autonomous and semi-autonomous systems. Connection Set Algebra has been developed to represent generative operations in neural networks but "none of these approaches are truly formal or general" (Jackson et al, 2017). There are some signs of progress, for example Selsam et al (2017) have used interactive proof assistants to implement ML algorithms and to state a formal theorem defining what it means for that implementation to be correct. Any residual implementation errors are detected because they would cause the proof to fail. Again, however, the approach has opnly been demonstrated on a small subset of the algorithms being used in safety-related systems. In this case they focus on the correctness of stochastic computation graphs with a machine-checkable proof that the gradients sampled by the system are unbiased estimates.

5 Conclusions and Further Work

Risk assessment techniques have evolved in response to changes in the demands of safety-related engineering. An increasing focus on the impact of operator intervention motivated the refinement of human reliability analysis. In parallel, the integration of software into control systems led to the development of standards that use proxies, including SILs, to avoid the explicit quantification of probability distributions for bugs that cannot be measured with any certainty.

We are facing new challenges to the continued use of risk assessment. Resurgent Artificial Intelligence and Machine Learning have matured to the point where they offer significant benefits across a host of safety-related applications.

Formal techniques are being developed to represent and reason about the behavior of systems that include AI based algorithms, these are in their infancy and none can, as yet, be integrated with traditional forms of risk assessment. In consequence, designers and regulators have very limited means of using deductive inference to demonstrate that this new generation of applications is acceptably safe. We also lack inductive tools to integrate these approaches into statistical forms of risk assessment. It is hard to anticipate how complex training sets will inform the detailed future operation of these applications when exposed to their eventual operating environment; past interactions may not provide clear predictions of future hazards.

Companies have responded to these concerns by developing safety arguments that assume significant constraints on the eventual context of use and that rely on longitudinal testing. These pragmatic approaches cannot sustain continued commercial development, especially in autonomous systems. There are fundamental questions about whether it is possible to maintain the segregation of these systems from more conventional vehicles as a means of simplifying the operating environment. Similarly, there is an urgent need for engineering and scientific foundations to the ‘proven in use’ approaches that are being exploited by a host of manufacturers that have abandoned existing software safety standards.

References

Authen, S. and Holmberg, J.-E., Reliability Analysis Of Digital Systems in a Probabilistic Risk Analysis For Nuclear Power Plants, Nuclear Engineering and Technology, (44)5:471-482

Butler, R. and G. Finelli, G. (1993), The Infeasibility of Quantifying the Reliability of Life-Critical Real-Time Software, IEEE Transactions on Software Engineering 19(1):3-12, 1993.

Ericson, C. (2016), Hazard Analysis techniques for System Safety, Second edition. John Wiley & Sons, NJ, USA.

Foord, A.G., Gulland, W.G. and Howard, C.R., (2011), Ten Years of IEC 61508; Has It Made Any Difference? Institute of Chemical Engineers, Symposium Series No. 156 Hazards XXII.

Greenland, S. (2001). Sensitivity Analysis, Monte Carlo Risk Analysis, and Bayesian Uncertainty Assessment, Journal of Risk Analysis, (21)4.

Hager, G.D., Bryant, R., Horvitz, E., Matarić, M, and Honavar, V. (2017), Advances in Artificial Intelligence Require Progress Across all of Computer Science, US National Science Foundation/Computing Community Consortium Catalyst, February. Available on: https://arxiv.org/pdf/1707.04352.pdf

Hollnagel, E. (1998) Cognitive Reliability and Error Analysis Method - CREAM. Elsevier

Jackson, E.C., Hughes, J.A., Daley, M. and Winter, M. (2017), An algebraic generalization for graph and tensor-based neural networks. Proceedings of the IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 23-25 Aug. 2017

Johnson, C.W., (2016), The Role of Regulators in Safeguarding the Interface between Autonomous Systems and the General Public. In J. Hewitt (ed.), Proceedings of the 34th International System Safety Conference, Orlando, USA 8-12 August 2016, International System Safety Society, Unionville, Virginia, USA.

Johnsen S. (2010) Resilience in Risk Analysis and Risk Assessment. In: Moore T., Shenoi S. (eds) Critical Infrastructure Protection IV. ICCIP 2010. IFIP Advances in Information and Communication Technology, vol 342. Springer, Berlin, Heidelberg.

Littlewood, B. and Strigini, L. (2013) 'The Validation of ultra-high dependability...' - 20 Years On, Safety Systems 20(3):6- 10.

Maguire, R., (2006), Safety Cases and Safety Reports, Method, Meaning and Motivation, CRC Press: Taylor and Francis, FL, USA.

Miller, K.W.; Morell, L.J., Noonan, R.E., Park, S.K., Nicol, D.M., Murrill, B.W. and Voas M. (1992). Estimating the probability of failure when testing reveals no failures, IEEE Transactions on Software Engineering (Volume: 18, Issue: 1, Jan.

Ostrom, L.T and Wilhelmsen (2012), C.A, Risk Assessment: Tools, Techniques and their Applications. John Wiley & Sons, NJ, USA.

Rausand, M. (2013), Risk Assessment: Theory, Method and Applications. John Wiley & Sons, NJ, USA.

Ramanathan, A., Pullum, L.L.,Hussain, F., Chakrabarty D., Jha, S.K. (2016), Integrating symbolic and statistical methods for testing intelligent systems: Applications to machine learning and computer vision. IEEE Design, Automation & Test in Europe Conference (DATE), Dresden, Germany, 2016

Reason, J. (2000). Human error: models and management, British Medical Journal, 320(7237): 768–770.

Salay, R., Quieroz, R., and Czarnecki, K. (2017) An Analysis of ISO 26262: Using Machine Learning Safely in Automotive Software. Available from https://arxiv.org/abs/1709.02435.

Saleh, J. and Marais, K. (2006), Highlights from the early (and pre-) history of reliability engineering, Reliability Engineering & System Safety, Volume 91, Issue 2, February, Pages 249-256.

Selsam, D., Liang, P. and Dill, D.L. (2017), Developing Bug-Free Machine Learning Systems With Formal Mathematics, Thirty-fourth International Conference on Machine Learning (ICML) 2017.

Swain A.D. and Guttman, H.E., 1983. Handbook of human reliability analysis with emphasis on nuclear power plant applications. US Nuclear Regulatory Commission), Washington, DC. NUREG/CR-1278.

UK Department for Transport (2015), The Pathway to Driverless Cars: Summary report and action plan, London, UK, February.

UK Health and Safety Executive (UK HSE, 1995), Out of Control, London, UK.

UK Health and Safety Executive (UK HSE, 2009), Review of Human Reliability Assessment Methods, London, UK.

UK Health and Safety Executive (UK HSE, 2001), Proposed Framework for Addressing Human Factors in IEC 61508, London, UK.

US Nuclear Regulatory Commission (US NRC, 1984), Probabilistic Risk Assessment (PRA) Reference Document: Final Report, Division of Risk Analysis and Operations, Office of Nuclear Regulatory Research, Washington DC, USA.

US Nuclear Regulatory Commission (US NRC, 1996), Technique for Human Event Analysis (ATHEANA) - Technical Basis and Methodological Description. NUREG/CR-6350. U.S. Nuclear Regulatory Commission, Brookhaven National Laboratory, Upton, NY, April. 1996. Prepared by Cooper, S.E., Ramey-Smith, A.M., Wreathall, J., Parry, G.W., Bley, D.C. Luckas, W.J., Taylor, J.H., Barriere, M.T.

Weiner, G. and B.W. Smith, B.W. (2016), Automated Driving: Legislative and Regulatory Action, Stanford University, USA. cybelaw.stanford.edu/wiki/index.php/, last accessed February 2016.

Williams, J.C. (1985). HEART – A Proposed Method for Achieving High Reliability in Elsevier, pp. 87-109.Process Operation by means of Human Factors Engineering Technology. In Proceedings of a Symposium on the Achievement of Reliability in Operating Plant, Safety and Reliability Society, 16 September 1985, Southport.