Efficient Explainability of Real-Time Schedulability

Sanjoy Baruah
Washington University in Saint Louis
baruah@wustl.edu

Pontus Ekberg
Uppsala University
pontus.ekberg@it.uu.se

***Abstract*—We had previously [1] proposed a rigorous definition of what it means for a safety property to be *efficiently explainable*: there should exist a certificate, whose validity may be checked by a deterministic algorithm in polynomial time, attesting to the safety of any system satisfying the property. Here we explore a more generalized notion of efficient explainability in which the validity of the certificate may be checked in a probabilistic sense (rather than only a deterministic one).**

***Index Terms*—Schedulability; Polynomial-time Verifiability; Randomized Verification.**

The first edition of this workshop threw up several alternative, all valid, interpretations of 'EXPLAINABILITY,' ranging from Andersson's rather informal perspective [2] that an explanation should be understandable to a non-expert, to far more formal interpretations such as the one articulated by Brandenburg during a panel discussion at the workshop that an explanation should be expressible in, and hence rigorously verifiable within, some machine-checkable formalism such as Prosa [3]. The authors of this submission had also provided a rigorous and formal perspective [1] as to what constitutes an acceptable explanation for a claim that a particular system satisfies a particular safety property; the central idea in [1] may be summarized as follows.

> By interpreting the set of all system specifications that satisfy a particular safety property as a *formal language* [4], and the explanation for a particular input system specification as a *certificate* attesting to the membership of that system specification in this formal language, the existence of explanations becomes closely related to several well-studied problems in computational complexity theory [5], [6]. In particular, explainable safety properties are exactly those for which the associated verification problems belong to well-defined complexity classes.

(From this perspective the Halting Problem [7] —the problem of determining, from a description of an arbitrary computer program P and an input e, whether the program will halt when executed upon this input— is explainable: for a given program P on input e, an acceptable certificate is simply the total number of steps that P executes on e before completing and halting. However, the complementary problem, that of determining whether P executes without halting on input e, is not explainable, as it is well-known that the complement of the halting problem is not recursively enumerable.)

This particular interpretation of explainability was investigated further in [8] by the authors of this submission, with a focus on *efficient* explainability: what are the safety properties for which there exist explanations that can be efficiently verified as being correct (or rejected for being erroneous – for failing to actually establish safety)? The central idea in [1], [8] can be summarized in the following proposition. Let us define a safety property to be *efficiently explainable* if for any system satisfying the safety property, there exists an explanation of this fact that can be validated for correctness by a deterministic procedure in time no worse than polynomial in the representation of the system; this definition simply equates efficient explainability with the complexity class NP.

Proposition 1.

- Any safety property for which the associated verification problem belongs to the complexity class NP is efficiently explainable; and
- Showing that the verification problem associated with a safety property is hard for a complexity class that is unlikely to be contained within NP offers strong evidence of that property not being efficiently explainable. □

The application of Proposition 1 was illustrated in our prior work [1], [8] upon several example problems concerning real-time schedulability analysis, including in particular (i) preemptive uniprocessor fixed-priority (FP) schedulability of constrained-deadline sporadic task systems (see, e.g., [9]–[11] for a description of this problem), and (ii) preemptive uniprocessor earliest-deadline-first (EDF) schedulability of sporadic task systems (this problem is described in, e.g., [12], [13]). It was pointed out that since FP-schedulability of constrained-deadline sporadic task systems is NP-complete [14], [15], it is in NP and hence efficiently explainable. In contrast, EDF-schedulability of sporadic task systems is known to be coNP-complete [16] and hence coNP-hard; since it is widely believed that coNP $\not\subseteq$ NP, this immediately implies that EDF-schedulability of sporadic task systems is unlikely to be efficiently explainable. In [8] this issue was addressed both via identifying efficiently explainable (NP) subsets of this coNP-hard problem, and via considering a couple of wider notions of efficient explainability, the latter in the form of pseudo-polynomial time verifiability (as captured by the class pseudoNP) and fully-polynomial time verification approximation schemes (FTPVAS). In this note we want to make the case for another intriguing possibility, the explainability of real-time schedulability using interactive proof systems.

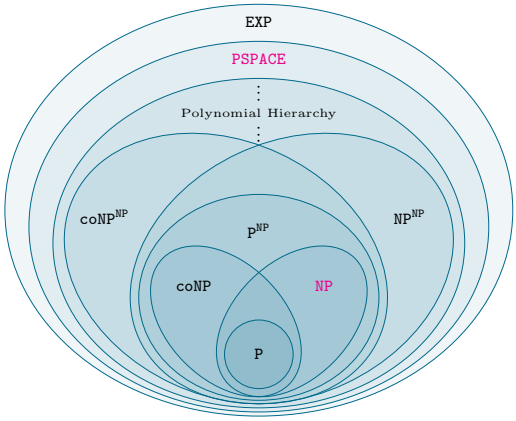

Fig. 1. Some computational complexity classes

RANDOMIZED VERIFICATION

Verification of safety-critical software has traditionally been a conservative endeavor, particularly when performed as part of a certification process in highly regulated domains such as civilian aviation or nuclear power control. It is interesting to speculate on the possibilities that open up if we were to settle for *randomized*, rather than purely deterministic, verification of safety claims. That is, rather than requiring, as current safety standards tend to do, that the correctness of a system be validated with absolute certainty, what if we would settle for a safety argument that convinces us of a system's safety at an arbitrarily high probability (that is strictly smaller than one — say, $(1 - 10^{-6})$)? If such randomized verification were to be considered acceptable, this opens up the possibility that rather than being a statically-generated certificate, an explanation be permitted to be of the form of an interactive dialog whereby a verifier makes repeated queries in order to develop adequate confidence in the veracity of a claim that a system satisfies a particular safety property. The reason why this would be a significant development builds upon a well-known result in complexity theory from the 1990's [17], establishing that the class of decision problems that can be verified in polynomial time by such an interactive randomized verifier (which communicates with a *prover* that is not polynomially bounded) is exactly the complexity class PSPACE. Hence if randomized interactive verification of safety properties were to be considered acceptable practice for the purposes of safety verification, then the class of efficiently explainable properties (i.e., the safety properties for which there exist polynomial-time verifiable interactive explanations for all systems satisfying the safety property) becomes the class of all safety properties for which the associated verification problem belongs to PSPACE. And as we can see in Figure 1, this complexity class is considerably larger than the class NP (which, recall, is the class of safety properties currently considered efficiently explainable – see Proposition 1). For instance, EDF-schedulability of sporadic task systems was shown to not be efficiently explainable under the prior definition (of deterministic verification); however since EDF-schedulability of sporadic task systems is coNP-complete and coNP $\subseteq$ PSPACE, it can be verified in polynomial time by a randomized interactive verifier. In a similar vein, global EDF- and FP-schedulability for sporadic task systems are both known to be in PSPACE [18], [19], and schedulability analysis for conditional DAG tasks is PSPACE-complete [20], [21]; hence such schedulability, too, can be verified in polynomial time by randomized interactive verifiers.

The possibility of interactive randomized verification of safety-critical systems becoming accepted practice opens up several interesting avenues of research, amongst them being the derivation of interactive proofs for important schedulability analysis problems that, in addition to having polynomially-bounded running time, are computationally reasonably efficient in practice. We point out that there are other, related, ongoing research efforts in this direction; see, e.g. [22].

REFERENCES

[1] S. Baruah and P. Ekberg, "Certificates of real-time schedulability," in *International Workshop on Explainability of Real-time Systems and their Analysis (ERSA)*, 2022.

[2] B. Andersson, "The case for explainability of real-time systems," 2022, presentation at International Workshop on Explainability of Real-time Systems and their Analysis (ERSA); PowerPoint slides available at https://sites.google.com/view/ersa22 (Date checked: 2023/ 09/13).

[3] F. Cerqueira, F. Stutz, and B. B. Brandenburg, "PROSA: A case for readable mechanized schedulability analysis," in *Proceedings of the 28th Euromicro Conference on Real-Time Systems (ECRTS)*, 2016, pp. 273–284.

[4] J. E. Hopcroft and J. D. Ullman, *Introduction To Automata Theory, Languages, And Computation*, 1st ed. USA: Addison-Wesley Longman Publishing Co., Inc., 1990.

[5] C. H. Papadimitriou, *Computational Complexity*. Addison-Wesley, 1994.

[6] S. Arora and B. Barak, *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. [Online]. Available: http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264

[7] A. Turing, "On computable numbers, with an application to the Entscheidungsproblem," in *Proceedings of the London Mathematical Society*, ser. 2, no. 42, 1936, pp. 230–265, correction in 43 (1937), pages 544–546.

[8] S. Baruah and P. Ekberg, "Towards Efficient Explainability of Schedulability Properties in Real-Time Systems," in *35th Euromicro Conference on Real-Time Systems (ECRTS 2023)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), A. V. Papadopoulos, Ed., vol. 262. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, pp. 2:1–2:20. [Online]. Available: https://drops.dagstuhl.de/opus/volltexte/2023/18031

[9] M. Joseph and P. Pandya, "Finding response times in a real-time system," *The Computer Journal*, vol. 29, no. 5, pp. 390–395, Oct. 1986.

[10] J. Lehoczky, L. Sha, and Y. Ding, "The rate monotonic scheduling algorithm: Exact characterization and average case behavior," in *Proceedings of the 10th Real-Time Systems Symposium (RTSS)*. Santa Monica, California, USA: IEEE Computer Society Press, Dec. 1989, pp. 166–171.

[11] N. C. Audsley, A. Burns, M. F. Richardson, and A. J. Wellings, "Hard Real-Time Scheduling: The Deadline Monotonic Approach," in *Proceedings 8th IEEE Workshop on Real-Time Operating Systems and Software*, Atlanta, May 1991, pp. 127–132.

[12] J. Y.-T. Leung and M. Merrill, "A note on the preemptive scheduling of periodic, real-time tasks," *Information Processing Letters*, vol. 11, pp. 115–118, 1980.

[13] S. Baruah, A. Mok, and L. Rosier, "Preemptively scheduling hard-real-time sporadic tasks on one processor," in *Proceedings of the 11th Real-Time Systems Symposium (RTSS)*. Orlando, Florida: IEEE Computer Society Press, 1990, pp. 182–190.

[14] F. Eisenbrand and T. Rothvoß, "Static-priority real-time scheduling: Response time computation is NP-hard," in *Proceedings of the 29th Real-Time Systems Symposium (RTSS)*. Barcelona: IEEE Computer Society Press, December 2008.

[15] P. Ekberg and W. Yi, "Fixed-priority schedulability of sporadic tasks on uniprocessors is NP-hard," in *Proceedings of the 38th Real-Time Systems Symposium (RTSS)*. IEEE Computer Society, 2017, pp. 139–146. [Online]. Available: https://doi.org/10.1109/RTSS.2017.00020

[16] F. Eisenbrand and T. Rothvoß, "EDF-schedulability of synchronous periodic task systems is coNP-hard," in *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, January 2010.

[17] A. Shamir, "IP = PSPACE," *Journal of the ACM*, vol. 39, no. 4, pp. 869–877, oct 1992. [Online]. Available: https://doi.org/10.1145/146585.146609

[18] V. Bonifaci and A. Marchetti-Spaccamela, "Feasibility analysis of sporadic real-time multiprocessor task systems," *Algorithmica*, vol. 63, no. 4, pp. 763–780, 2012. [Online]. Available: https://doi.org/10.1007/s00453-011-9505-6

[19] G. Geeraerts, J. Goossens, and M. Lindström, "Multiprocessor schedulability of arbitrary-deadline sporadic tasks: complexity and antichain algorithm," *Real Time Systems*, vol. 49, no. 2, pp. 171–218, 2013. [Online]. Available: https://doi.org/10.1007/s11241-012-9172-y

[20] S. Baruah and A. Marchetti-Spaccamela, "Feasibility Analysis of Conditional DAG Tasks," in *Proceedings of the 33rd Euromicro Conference on Real-Time Systems (ECRTS)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), B. B. Brandenburg, Ed., vol. 196. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, pp. 12:1–12:17. [Online]. Available: https://drops.dagstuhl.de/opus/volltexte/2021/13943

[21] ——, "The computational complexity of feasibility analysis for conditional DAG tasks," *ACM Trans. Parallel Comput.*, jul 2023. [Online]. Available: https://doi.org/10.1145/3606342

[22] E. Couillard, P. Czerner, J. Esparza, and R. Majumdar, "Making IP = PSPACE practical: Efficient interactive protocols for BDD algorithms," in *Computer Aided Verification*, C. Enea and A. Lal, Eds. Cham: Springer Nature Switzerland, 2023, pp. 437–458.