Safety and Liveness Properties: A Survey

Ekkart Kindler*

Institut für Informatik der
Technischen Universität München
Arcisstraße 21
D-80333 München
Germany

and

Humboldt-Universität zu Berlin
Institut für Informatik
Unter den Linden 6
D-10099 Berlin
Germany

Abstract

The distinction of safety and liveness properties is often adopted in specification and design methods for distributed systems. We present a short survey on the "history" of these concepts and on papers that contributed to their general acceptance.

The notions of *safety* and *liveness* properties have been first introduced by Lamport [14]. Informally, a safety property expresses that "something (bad) will *not* happen" during a system execution. A liveness property expresses that eventually "something (good) *must* happen" during an execution. The distinction of safety and liveness properties was motivated by the different techniques for proving those properties. For example, Owicki and Lamport [16] propose the technique of proof lattices for liveness properties. Later, Lamport makes his informal characterization of safety properties more precise [4]. An *execution* of a distributed system is formalized as an infinite sequence of states. Any set of such sequences is a *property*. A property is called a safety property (Section 2.2 in [4]), if and only if each execution violating the property has a finite prefix[1] violating that property and, vice versa[2], if a finite prefix of an execution violates the property then the execution itself violates the property. This corresponds to the intuition that the "bad thing" (i.e. violating the property) can be detected in a finite initial part of the execution and the occurrence of the "bad thing" in a prefix of an execution is irremediable.

The notion of safety properties is also convincing because a safety property can be *generated* by a transition systems with finite internal nondeterminism [3]. This "property of safety properties" seems to be one of the main justifications for the adequacy of the

*supported by the Deutsche Forschungsgemeinschaft SFB 342, TP A3: SEMAFOR and the ESPRIT Basic Research WG 6067 Caliban

[1] Since Lamport considers only infinite sequences he needs a "trick" to talk about finite executions: the last state is repeated infinitely often.

[2] In Section 2.2 of [4] Lamport requires only the first direction; Schneider et al. (Section 5.2 of [4] and [6, 5]) associate Lamport also with the definition of safety requiring both directions.

definition of safety properties. To our knowledge this result was for the first time formally stated by Abadi and Lamport [3] (Proposition 3).

Alpern and Schneider [6] were the first to give a formal definition of both safety and liveness properties. In contrast to Lamport [4] they represent a finite prefix of an execution as the set of all possible continuations from that point on, which leads to a slightly more general notion of safety properties.[3] A property is a liveness property, if and only if it contains at least one continuation for every finite prefix. This corresponds to the intuition that the "good thing" (i.e. satisfying the property) can still happen after any finite execution.

It turns out that safety properties are closed under conjunction and finite disjunction. Therefore, the characterization of safety and liveness properties corresponds to a topology, where safety properties are the closed sets and liveness properties are the dense sets. This correspondence was used [6] to prove that every property can be represented as the conjunction of a safety and a liveness property (*decomposition theorem*).[4] The *safety part* of the decomposition of a property P is the least safety property containing P, which corresponds to the closure $\overline{P}$ of topology. Later on Schneider [18] presents a proof of the decomposition theorem without (explicitly) using topological arguments. Moreover, Alpern and Schneider discuss [6] some other characterizations of liveness properties, which have not got generally adopted.

Due to the different representation of finite prefixes, Lamport's and Alpern-Schneider's characterizations of safety properties are not equivalent. Alpern, Demers, and Schneider [5] motivate their definition by a property that is a safety property according to their definition, but not according to Lamport's: "The value of a counter increases in every step"[5]. The reason for this difference is that this property is not *invariant under stuttering* (insertion of idling steps in an execution). For properties that are invariant under stuttering, however, both definitions coincide [5]. Moreover, an explicit use of finite executions for the characterization of safety and liveness properties (e.g. [10]) settles the matter.

Sistla [19] characterizes the structure of temporal formulas that represent safety and liveneness properties (according to [6]), respectively. Moreover, he introduces the notions of *strong safety* and *absolute liveness properties*[6] and characterizes the structure of the corresponding temporal formulas. In [7] safety and liveness properties are characterized by means of structural properties of Büchi automata. The decomposition theorem can be interpreted as a transformation of the corresponding Büchi automaton. Rem [17] gives a characterization of safety and liveness properties equivalent[7] to the characterization

[3] [4] and [6] consider only infinite executions. [10] considers finite and infinite executions, which results in a more elegant characterization of safety and liveness properties.

[4] In addition, they show that every property can also be represented as the conjunction of two liveness properties.

[5] Provided that only infinite executions are regarded, this property can be considered as a safety property.

[6] [19, 6] have been previously published as technical reports, and thus contain mutual references. The characterization of safety and liveness properties seems to be due to Alpern and Schneider, strong safety and absolute liveness seems to be due to Sistla.

[7] Except for a different mathematical presentation no new aspects are added.

of Alpern and Schneider. His characterization is based on an equivalence relation on properties, where two properties are equivalent, if they have the same set of finite prefixes. Then, every safety property is the greatest element of an equivalence class and every liveness property is equivalent to the set of all executions.

Lichtenstein, Pnueli, and Zuck [15] present a characterization of safety and liveness properties that is based on the syntactic structure of temporal formulas representing the property. In contrast to [19] their temporal logic contains *past-operators*. Their characterization of safety properties coincides with the one of Alpern and Schneider [6] (as long as the property is representable by a temporal formula), the characterization of liveness properties, however, is more general. For example, until-properties, which are no liveness properties according to [6] and which have a non-trivial safety part, are regarded as liveness properties. Therefore, the nice correspondence to topology is spoilt.

Not only theory (decomposition theorem) but also practise (different proof techniques) show that properties should be specified as a conjunction of a safety and a liveness property. Sometimes this decomposition of specifications results in an unexpected effect [8, 10, 3, 13]: a transition system generating the safety part can get stuck without being able to establish the liveness part. It is, so to speak, "painting itself into a corner" [8]. A decomposition that avoids this "anomaly" [10] is called *feasible* [8], *live with respect to a safety property* [10], *machine closed* [3] or *congruous* [13]. The idea of this notion is that the decomposition is *operational* (cf. [13]). Therefore, Dederichs and Weber [10] propose *always* to specify liveness with respect to a safety property. Non-machine-closed specifications should be avoided. Several authors [1] reply that this is not desirable in general. Only on the lower (implementation) level the requirement of machine closedness is sensible. In reply, Dederichs and Weber [11] state that a machine closed decomposition exists, at least (cf. [13]).

Abadi and Lamport [2] consider the notion of a machine closed decomposition in a different contex; there, the notion of a machine closed specification becomes a more extensive meaning with respect to compositional design methods. Henzinger [13] investigates the notion of a decomposition of a property with respect to another property in more detail; he is especially concerned in realtime properties.

Gumm approaches the concept of safety and liveness properties from a different direction [12]: he investigates the necessary conditions for the decomposition theorem [6]. Then he generalizes the notions of safety and liveness properties accordingly. It remains an open question whether there are proof techniques corresponding to his characterization and with that whether his characterization has practical relevance.

A quite different kind of classification was proposed by Chang, Manna, and Pnueli [9]. They do not characterize a partition but a hierachie of properties. The least level of this hierachy coincides with the characterization of safety properties. The higher levels provide a finer distinction of different liveness properties. Their approach is likewise motivated by the different techniques for proving the properties of the different levels. A detailed discussion of this hierachy is out of the scope of this paper.

References

[1] M. Abadi, B. Alpern, K.R. Apt, N. Francez, S. Katz, L. Lamport, and F.B. Schneider. Preserving liveness: Comments on "Safety and liveness from a methodological point of view". *Information Processing Letters*, 40:141–142, November 1991.

[2] Martín Abadi and Leslie Lamport. Composing specifications. In J.W. de Bakker and W.-P. de Roever, editors, *Stepwise Refinement of Distributed Systems. Models, Formalisms, Correctness.*, *LNCS* 430. Springer-Verlag, 1990. previously: SRC Research Report 66, October 1990.

[3] Martín Abadi and Leslie Lamport. The existence of refinement mappings. *Theoretical Computer Science*, 82:253–284, 1991. previously: SRC Research Report 27, April 1988.

[4] M.W. Alford, J.P. Ansart, G. Hommel, L. Lamport, B. Liskov, G.P. Mullery, and F.B. Schneider. *Distributed Systems: Methods and Tools for Specification*, *LNCS* 190. Springer-Verlag, 1985.

[5] Bowen Alpern, Alan J. Demers, and Fred B. Schneider. Safety without stuttering. *Information Processing Letters*, 23:177–180, November 1986. previously: Technical Report 85-708, Cornell University, October 1985.

[6] Bowen Alpern and Fred B. Schneider. Defining liveness. *Information Processing Letters*, 21:181–185, October 1985.

[7] Bowen Alpern and Fred B. Schneider. Recognizing safety and liveness. Technical Report 86-727, Cornell University, Ithaca, New York, January 1986.

[8] Krysztof R. Apt, Nissim Francez, and Shmuel Katz. Appraising fairness in languages for distributed programming. *Distributed Computing*, 2:226–241, 1988.

[9] Edward Chang, Zohar Manna, and Amir Pnueli. The safety-progress classification. In F.L. Bauer, W. Brauer, and H. Schwichtenberg, editors, *Logic and Algebra of Specifications*, *NATO ASI Series F: Computer and Systems Sciences* Vol. 94, pages 143–202. Springer-Verlag, 1993. previously: Technical Report STAN-CS-92-1408, February 1992.

[10] Frank Dederichs and Rainer Weber. Safety and liveness from a methodological point of view. *Information Processing Letters*, 36:25–30, October 1990.

[11] Frank Dederichs and Rainer Weber. Reply to the comments by M. Abadi et al. *Information Processing Letters*, 40:143, November 1991.

[12] H. Peter Gumm. Another glance at the Alpern-Schneider characterization of safety and progress in concurrent executions. Preliminary version of an article, received 1993.

[13] Thomas A. Henzinger. Sooner is safer than later. *Information Processing Letters*, 43:135–141, September 1992.

[14] Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Transactions on Software Engineering*, SE-3(2):125–143, March 1977.

[15] Orna Lichtenstein, Amir Pnueli, and Lenore Zuck. The glory of the past. In Rohit Parikh, editor, *Logics of Programs*, *LNCS* 193, pages 196–218. Springer-Verlag, June 1985.

[16] Susan Owicki and Leslie Lamport. Proving liveness properties of concurrent programs. *ACM Transactions on Programming Languages and Systems*, 4(3):455–495, July 1982.

[17] Martin Rem. A personal perspective of the Alpern-Schneider characterization of safety and liveness. In W.H.J. Feijen, A.J.M van Gasteren, D. Gries, and J. Misra, editors, *Beauty Is Our Business*, Texts and Monographs in Computer Science, chapter 43, pages 365–372. Springer-Verlag, 1990.

[18] Fred B. Schneider. Decomposing properties into safety and liveness using predicate logic. Technical Report 87-874, Department of Computer Science, Cornell University, Ithaca, New York, October 1987.

[19] A.P. Sistla. On the characterization of safety and liveness properties in temporal logic. In *Proceedings of the 4^{th} annual ACM Symposium on Principles of Distributed Computing*, pages 39–48, Minaki, Ontario, Canada, August 1985. ACM.