Ahrendt, Wolfgang; **Chimento, Jesús Mauricio**; **Pace, Gordon J.**; **Schneider, Gerardo**
Verifying data- and control-oriented properties combining static and runtime verification: theory and tools. (English) Zbl 1370.68195
Form. Methods Syst. Des. 51, No. 1, 200-265 (2017).

Summary: Static verification techniques are used to analyse and prove properties about programs before they are executed. Many of these techniques work directly on the source code and are used to verify data-oriented properties over all possible executions. The analysis is necessarily an over-approximation as the real executions of the program are not available at analysis time. In contrast, runtime verification techniques have been extensively used for control-oriented properties, analysing the current execution path of the program in a fully automatic manner. In this article, we present a novel approach in which data-oriented and control-oriented properties may be stated in a single formalism amenable to both static and dynamic verification techniques. The specification language we present to achieve this that of ppDATEs, which enhances the control-oriented property language of DATEs, with data-oriented pre/postconditions. For runtime verification of ppDATE specifications, the language is translated into a DATE. We give a formal semantics to ppDATEs, which we use to prove the correctness of our translation from ppDATEs to DATEs. We show how ppDATE specifications can be analysed using a combination of the deductive theorem prover KeY and the runtime verification tool LARVA. Verification is performed in two steps: KeY first partially proves the data-oriented part of the specification, simplifying the specification which is then passed on to LARVA to check at runtime for the remaining parts of the specification including the control-oriented aspects. We show the applicability of our approach on two case studies.

MSC:

68Q60 Specification and verification (program logics, model checking, etc.)
68N19 Other programming paradigms (object-oriented, sequential, concurrent, automatic, etc.)
68N30 Mathematical aspects of software engineering (specification, verification, metrics, requirements, etc.)

Keywords:

runtime verification; static verification; Java; program analysis

Software:

LARVA; MarQ; KeY; StaRVOOrS; JUnit; DyTa; ESC/Java; VeriFast; Java-MOP; Clara; Pex; Z; Dafny; Spec; JML

Full Text: DOI

References:

[1] Apache Tomcat. http://tomcat.apache.org/

[2] Ahrendt W, Beckert B, Bubel R, Hähnle R, Schmitt PH, Ulbrich M (eds) (2016) Deductive software verification—the KeY book (LNCS), vol 10001. Springer, Berlin · Zbl 1080.68062

[3] Ahrendt W, Chimento JM, Pace GJ, Schneider G (2015) A specification language for static and runtime verification of data and control properties. In: FM'15 (LNCS), vol 9109. Springer, Berlin

[4] Ahrendt, W; Dylla, M, A system for compositional verification of asynchronous objects, Sci Comput Program, 77, 1289-1309, (2012) · Zbl 1264.68050 · doi:10.1016/j.scico.2010.08.003

[5] Ahrendt W, Pace G, Schneider G (2012) A unified approach for static and runtime verification: framework and applications. In: ISoLA'12 (LNCS), vol 7609. Springer, Berlin

[6] Ahrendt W, Pace GJ, Schneider G (2016) StaRVOOrS—episode II: strengthen and distribute the force. In: ISoLA'16 (1) (LNCS), vol 9952. Springer, Berlin · Zbl 1080.68062

[7] Artho, C; Barringer, H; Goldberg, A; Havelund, K; Khurshid, S; Lowry, M; Pasareanu, C; Rosu, G; Sen, K; Visser, W; etal., Combining test case generation and runtime verification, Theor Comput Sci, 336, 209-234, (2005) · Zbl 1080.68062 · doi:10.1016/j.tcs.2004.11.007

[8] Artho C, Biere A (2015) Combined static and dynamic analysis. In: AIOOL'05 (ENTCS) vol 131, pp 3-14

[9] Barnes J (2012) SPARK: the proven approach to high integrity software. Altran Praxis. http://www.altran.co.uk

[10] Barnett M, Rustan K, Leino M, Schulte W (2005) The Spec# programming system: an overview. In: CASSIS'05 (LNCS) vol 3362. Springer, Berlin, pp 49-69

[11] Barringer H, Goldberg A, Havelund K, Sen K (2004) Rule-based runtime verification. In: VMCAI'04, pp 44-57 · Zbl 1202.68243

[12] Bodden E, Hendren LJ, Lhoták O (2007) A staged static program analysis to improve the performance of runtime monitoring. In: ECOOP'07 (LNCS), vol 4609

[13] Bodden E, Lam P (2010) Clara: partially evaluating runtime monitors at compile time—tutorial supplement. In: RV'10 (LNCS) vol 6418, pp 74-88

[14] Burdy, L; Cheon, Y; Cok, DR; Ernst, MD; Kiniry, JR; Leavens, GT; Rustan, K; Leino, M; Poll, E, An overview of JML tools and applications, Int J Softw Tools Technol Transf, 7, 212-232, (2005)· doi:10.1007/s10009-004-0167-4

[15] Chen F, Roşu G (2005) Java-MOP: a monitoring oriented programming environment for Java. In: TACAS'05 (LNCS), vol 3440. Springer, Berlin, pp 546-550 · Zbl 1087.68550

[16] Chimento JM, Ahrendt W, Pace GJ, Schneider G (2015) StaRVOOrS: a tool for combined static and runtime verification of Java. In: Bartocci E, Majumdar R (eds) Runtime verification (LNCS), vol 9333. Springer, Berlin, pp 297-305

[17] Christakis M, Müller P, Wüstholz V (2012) Collaborative verification and testing with explicit assumptions. In: FM'12: formal methods - 18th international symposium, Paris, France, August 27-31, 2012. Proceedings, pp 132-146

[18] Colombo C, Pace GJ, Schneider G (2009) Dynamic event-based runtime monitoring of real-time and contextual properties. In: FMICS'08 (LNCS), vol 5596. Springer, Berlin, pp 135-149

[19] Colombo C, Pace GJ, Schneider G (2009) LARVA: a tool for runtime monitoring of Java programs. In: SEFM'09, IEEE Computer Society, pp 33-37

[20] Csallner C, Smaragdakis Y(2005) Check 'n' crash: combining static checking and testing. In: 27th International Conference on Software Engineering (ICSE 2005), 15-21 May 2005, St. Louis, Missouri, USA, pp 422-431

[21] de Boer FS, de Gouw S, Johnsen EB, Wong PYH (2013) Run-time checking of data- and protocol-oriented properties of Java programs: an industrial case study. In: Shin Sung Y, Maldonado Jos C (eds) SAC. ACM, pp 1573-1578

[22] Decker N, Leucker M, Thoma D (2013) jUnitRV—adding runtime verification to JUnit. In: NASA formal methods (LNCS), vol 7871. Springer, Berlin

[23] Ernst, G; Pfähler, J; Schellhorn, G; Haneberg, D; Reif, W, KIV: overview and verifythis competition, Int J Softw Tools Technol Transf, 17, 677-694, (2015)· doi:10.1007/s10009-014-0308-3

[24] Falzon K, Pace G (2012) Combining testing and runtime verification techniques. In Model-based methodologies for pervasive and embedded software, 8th international workshop, MOMPES 2012, Essen, Germany, September 4, 2012, pp 38-57

[25] Flanagan Cormac, Leino K Rustan M, Lillibridge Mark, Nelson Greg, Saxe James B, Stata Raymie (2002) Extended Static Checking for Java. In Knoop Jens, Hendren Laurie J , editors, PLDI'02, pages 234-245. ACM

[26] Ge X, Taneja K, Xie T, Tillmann N (2011) DyTa: dynamic symbolic execution guided with static verification results. In: Proceedings of the 33rd international conference on software engineering, ICSE 2011, Waikiki, Honolulu , HI, USA, May 21-28, 2011, pp 992-994

[27] Gries D (1987) The science of programming, 1st edn. Springer, Berlin · Zbl 0614.68002

[28] Jacobs B, Smans J, Philippaerts P, Vogels F, Penninckx W, Piessens F (2011) Verifast: a powerful, sound, predictable, fast verifier for C and Java. In: NASA formal methods (LNCS), vol 6617. Springer, pp 41-55

[29] Leavens GT, Poll E, Clifton C, Cheon Y, Ruby C, Cok D, Müller P, Kiniry J, Chalin P (2007) JML reference manual. Draft 1.200

[30] Leino K Rustan M (2010) Dafny: an automatic program verifier for functional correctness. In: Clarke EM, Voronkov A (eds) Logic for programming, artificial intelligence, and reasoning (LPAR-16) (LNCS), vol 6355. Springer, Berlin · Zbl 1253.68095

[31] Maraninchi F, Rémond Y (2000) Running-modes of real-time systems: a case-study with mode-automata. In: Proceedings of 12th euromicro conference on real-time systems (ECRTS 2000), 19-21 June 2000, Stockholm, Sweden, pp 257-264

[32] MasterCard International Inc. Mondex web page. http://www.mondexusa.com/

[33] Reger G (2016) An overview of MarQ. In: Proceedings of runtime verification—16th international conference, RV 2016 (LNCS), vol 10012. Springer

[34] Sözer, H, Integrated static code analysis and runtime verification, Softw Pract Exp, 45, 1359-1373, (2015)· doi:10.1002/spe.2287

[35] Spivey JM (1989) The Z notation: a reference manual. Prentice-Hall Inc, Upper Saddle River · Zbl 0777.68003

[36] SoftSlate Commerce. www.softslate.com/

[37] Stepney S, Cooper D, Woodcock J (2000) An electronic purse: specification, refinement and proof. Technical monograph PRG-126, Oxford University Computing Laboratory

[38] StaRVOOrS web page. http://cse-212294.cse.chalmers.se/starvoors/

[39] Tillmann N, Halleux Jonathan de (2008) Pex-white box test generation for .nET. In: Beckert B, Hähnle R (eds) Tests and proofs (LNCS), vol 4966. Springer, Berlin, pp 134-153

[40] Tonin I (2007) Verifying the mondex case study. The KeY approach. Technical Report 2007-4, Universität Karlsruhe

[41] Wonisch D, Schremmer A, Wehrheim H (2013) Zero overhead runtime monitoring. In: SEFM'13 (LNCS), vol 8137. Springer,

Berlin, pp 244-258

[42] Woodcock J (2006) First steps in the verified software grand challenge. In: SEW'06. IEEE Computer Society, pp 203-206

[43] Zee K, Kuncak V, Taylor M, Rinard MC (2007) Runtime checking for program verification. In: RV'07 (LNCS), vol 4839. Springer, Berlin, pp 202-213