CPAchecker

Marian Lingsch-Rosenfeld
Daniel Baier, Dirk Beyer, Marek Jankola, Matthias Kettl, Philipp Wendler

May 5, 2025
LMU Munich, SoSy-Lab

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

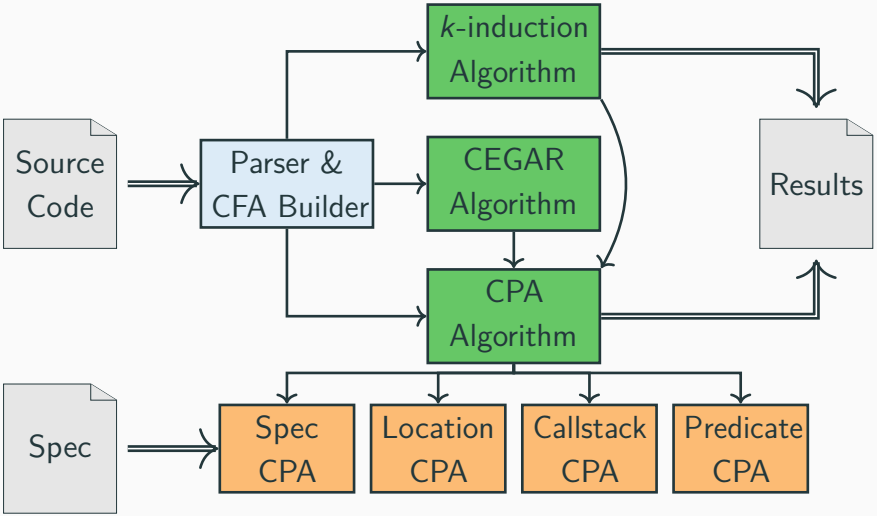
SoSy-Lab Software Systems

Overview

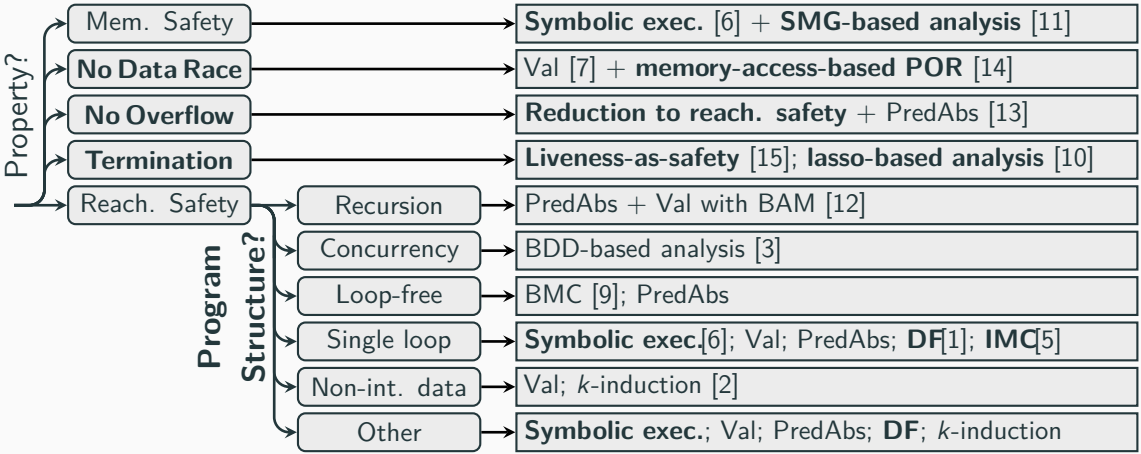
CPAchecker is a modern and versatile framework for building software-verification and witness-validation analyses.

- Supports all SV-COMP properties and categories of C programs
- Can export and validate all witness types and formats
- Utilizes many different algorithms

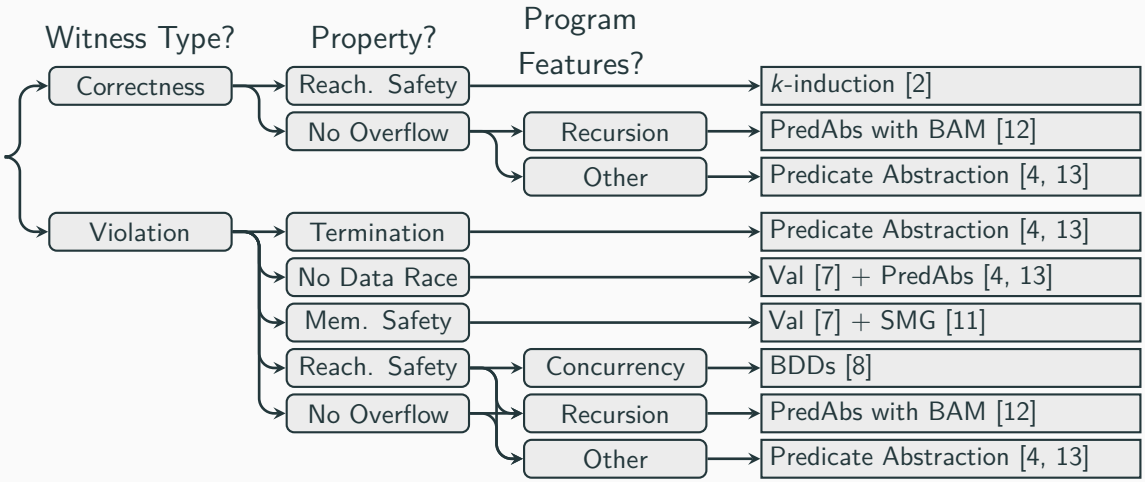
Architecture

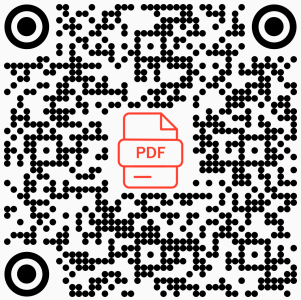


Strategy Selection: Verification



Strategy Selection: Validation

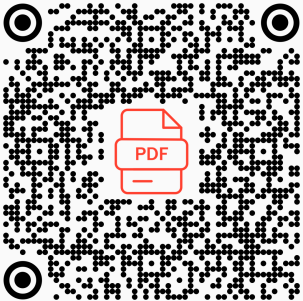




CPAchecker tutorial paper

New this year:

- Parallel portfolio of analyses
- Improved memory analysis
- Disabling export of witnesses v1.0



CPAchecker validator paper

References i

[1] Beyer, D., Chien, P.C., Lee, N.Z.: CPA-DF: A tool for configurable interval analysis to boost program verification. In: Proc. ASE. pp. 2050–2053. IEEE (2023). https://doi.org/10.1109/ASE56229.2023.00213

[2] Beyer, D., Dangl, M., Wendler, P.: Boosting k-induction with continuously-refined invariants. In: Proc. CAV. pp. 622–640. LNCS 9206, Springer (2015). https://doi.org/10.1007/978-3-319-21690-4_42

[3] Beyer, D., Friedberger, K.: A light-weight approach for verifying multi-threaded programs with CPAchecker. In: Proc. MEMICS. vol. 233, pp. 61–71. EPTCS (2016). https://doi.org/10.4204/EPTCS.233.6

References ii

[4] Beyer, D., Keremoglu, M.E., Wendler, P.: Predicate abstraction with adjustable-block encoding. In: Proc. FMCAD. pp. 189–197. FMCAD (2010), https://www.sosy-lab.org/research/pub/2010-FMCAD.Predicate_Abstraction_with_Adjustable-Block_Encoding.pdf

[5] Beyer, D., Lee, N.Z., Wendler, P.: Interpolation and SAT-based model checking revisited: Adoption to software verification. arXiv/CoRR **2208**(05046) (July 2022). https://doi.org/10.48550/arXiv.2208.05046

[6] Beyer, D., Lemberger, T.: CPA-SymExec: Efficient symbolic execution in CPAchecker. In: Proc. ASE. pp. 900–903. ACM (2018). https://doi.org/10.1145/3238147.3240478

References iii

[7] Beyer, D., Löwe, S.: Explicit-state software model checking based on CEGAR and interpolation. In: Proc. FASE. pp. 146–162. LNCS 7793, Springer (2013). https://doi.org/10.1007/978-3-642-37057-1_11

[8] Beyer, D., Stahlbauer, A.: BDD-based software model checking with CPAchecker. In: Proc. MEMICS. pp. 1–11. LNCS 7721, Springer (2013). https://doi.org/10.1007/978-3-642-36046-6_1

[9] Biere, A., Cimatti, A., Clarke, E.M., Zhu, Y.: Symbolic model checking without BDDs. In: Proc. TACAS. pp. 193–207. LNCS 1579, Springer (1999). https://doi.org/10.1007/3-540-49059-0_14

References iv

[10] Cook, B., Podelski, A., Rybalchenko, A.: TERMINATOR: Beyond safety. In: Proc. CAV. pp. 415–418. LNCS 4144, Springer (2006). https://doi.org/10.1007/11817963_37

[11] Dudka, K., Peringer, P., Vojnar, T.: Byte-precise verification of low-level list manipulation. In: Proc. SAS. pp. 215–237. LNCS 7935, Springer (2013). https://doi.org/10.1007/978-3-642-38856-9_13

[12] Friedberger, K.: CPA-BAM: Block-abstraction memoization with value analysis and predicate analysis (competition contribution). In: Proc. TACAS. pp. 912–915. LNCS 9636, Springer (2016). https://doi.org/10.1007/978-3-662-49674-9_58

References v

[13] Henzinger, T.A., Jhala, R., Majumdar, R., McMillan, K.L.: Abstractions from proofs. In: Proc. POPL. pp. 232–244. ACM (2004). https://doi.org/10.1145/964001.964021

[14] Peled, D.: Ten years of partial order reduction. In: Proc. CAV. pp. 17–28. Springer (1998). https://doi.org/10.1007/BFb0028727

[15] Schuppan, V., Biere, A.: Liveness checking as safety checking for infinite state spaces. Electr. Notes Theor. Comput. Sci. **149**(1), 79–96 (2006). https://doi.org/10.1016/j.entcs.2005.11.018