

# Tools for CSP

Markus Roggenbach\*  
Department of Computer Science  
Swansea University  
Swansea, Wales, United Kingdom  
csmarkus@swan.ac.uk



Among the various frameworks for the description and modelling of reactive systems, process algebra [2] plays a prominent role. It has proved to be suitable, e.g., at the level of requirement specification, at the level of design specification, and also for the formal analysis of systems. In this context, the process algebra CSP [10, 22, 11] provides a well-established, theoretically thoroughly studied, and in industry applied formalism for the modelling and verification of concurrent systems. Its applications range from train control systems, see e.g. [5, 9], over software for the international space station [3, 4] to the verification of security protocols, see e.g. [23]. In 2005, the CSP community celebrated its birthday with the workshop "Communicating Sequential Processes, the first 25 years" [1].

From its beginning on, CSP came along with simulation techniques. Hoare's book "Communicating Sequential Processes" [10] from 1985 devotes separate sections on how to implement a simulation of the CSP semantics in a functional programming language. The tool PROBE is one of many simulators available. Concerning proofs, CSP originally started as a new branch of mathematics where proofs are to be carried out with paper and pen. Soon enough, however, model checking was developed for CSP, offering a fully automated proof technique. This endeavor has been so successful that the CSP model checker FDR [19] has been available as a commercial product for more than 20 years.

Over the last decade interactive theorem proving tools also have become available for CSP, complementing model checking techniques. The aim here is to ease the analysis of infinite state systems. A first pioneer here was Camilleri in as early as 1990 [6]. In 1997, Dutertre and Schneider [8] presented a CSP encoding based on the theorem prover PVS [24]. Their tool is tailored to the analysis of security

protocols. With such theorem proving support it is also possible to directly analyze entire classes of systems. Taking up the encoding of Dutertre and Schneider, Wei and Heather give in 2005, e.g., a mechanized proof for the deadlock freedom of the dining philosophers problem independent of the number of philosophers involved [26]. Yet another result of this type is the mechanized proof that a certain family of systolic arrays is deadlock-free, independent of the number of processes involved [18], which Gruner, Isobe and Roggenbach presented in 2005.

Another effect of encoding CSP into theorem provers is that formalizing CSP can help to reveal flaws in the semantic construction of the language itself. In 1997 Wolff and Tej [25] published a corrected Failure-Divergence model for CSP, which arose from an implementation of CSP using the theorem prover Isabelle/HOL [20]. While Wolff and Tej use a so-called shallow encoding, Isobe and Roggenbach presented in 2005 the tool CSP-Prover [12, 13, 14, 15, 16, 17, 18], which provides a deep encoding of CSP in Isabelle/HOL. This kind of encoding allows one to reflect on the language CSP itself. In 2006 Isobe and Roggenbach proved within CSP-Prover the completeness of an axiom system for the CSP stable failures model. This proof also lead to the correction of algebraic laws [14]. In the perception of the CSP community such results demonstrate that presentations of CSP models and CSP axiom schemes must necessarily be accompanied by mechanized theorem proving tools.

The talk will discuss various proof tools for CSP, show examples of how to analyze systems using these tools, and will also cover recent developments such as the tool HORAE [7], which is based on constraint satisfaction techniques, and the CSP-CASL-Prover [21], which provides integrated theorem proving for processes and data.

**Acknowledgment** The author would like to thank Y Isobe for the productive and highly amicable research cooperation on CSP-Prover and Erwin R Catesbeiana (Jr) for inspiration on parallel processing.

\*Sponsored by Formal Methods Europe (FME).

## References

- [1] A. Abdallah, C. Jones, and J. Sanders, editors. *Communicating Sequential Processes: The First 25 Years*. LNCS 3525. Springer, 2005.
- [2] J. A. Bergstra, A. Ponse, and S. A. Smolka, editors. *Handbook of Process Algebra*. Elsevier, 2001.
- [3] B. Buth, M. Kouvaras, J. Peleska, and H. Shi. Deadlock analysis for a fault-tolerant system. In M. Johnson, editor, *AMAST'97*, LNCS 1349, pages 60–75. Springer, 1997.
- [4] B. Buth, J. Peleska, and H. Shi. Combining methods for the livelock analysis of a fault-tolerant system. In A. M. Haeberer, editor, *AMAST'98*, LNCS 1548, pages 124–139. Springer, 1998.
- [5] B. Buth and M. Schröner. Model-checking the architectural design of a fail-safe communication system for railway interlocking systems. In J. M. Wing, J. Woodcock, and J. Davies, editors, *FM'99*, LNCS 1709. Springer, 1999.
- [6] A. J. Camilleri. Mechanizing CSP trace theory in higher order logic. *IEEE Transactions on Software Engineering*, 16(9):993–1004, 1990.
- [7] J. S. Dong, P. Hao, J. Sun, and X. Zhang. A reasoning method for timed CSP based on constraint solving. In Z. Liu and H. Jifeng, editors, *Formal Methods and Software Engineering*, LNCS 4260, pages 342–359. Springer, 2006.
- [8] B. Dutertre and S. Schneider. Using a PVS embedding of CSP to verify authentication protocols. In E. L. Gunter and A. Felty, editors, *TPHOL 1997*, LNCS 1275, pages 121–136. Springer, 1997.
- [9] A. E. Haxthausen and J. Peleska. A domain-oriented, model-based approach for construction and verification of railway control systems. In C. B. Jones, Z. Liu, and J. Woodcock, editors, *Formal Methods and Hybrid Real-Time Systems*, LNCS 4700, pages 320–348. Springer, 2007.
- [10] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.
- [11] T. Hoare. Why ever CSP? *Electronic Notes in Theoretical Computer Science*, 162:209–215, 2006.
- [12] Y. Isobe and M. Roggenbach. Webpage on CSP-Prover. <http://staff.aist.go.jp/y-isobe/CSP-Prover/CSP-Prover.html>.
- [13] Y. Isobe and M. Roggenbach. A generic theorem prover of CSP refinement. In N. Halbwachs and L. D. Zuck, editors, *TACAS 2005*, LNCS 3440, pages 108–123. Springer, 2005.
- [14] Y. Isobe and M. Roggenbach. A complete axiomatic semantics for the CSP stable-failures model. In C. Baier and H. Hermanns, editors, *CONCUR 2006*, LNCS 4137, pages 158–172. Springer, 2006.
- [15] Y. Isobe and M. Roggenbach. CSP-Prover - a proof tool for the verification of scalable concurrent systems. *Japan Society for Software Science and Technology Computer Software*, 25, 2008.
- [16] Y. Isobe and M. Roggenbach. Proof Principles of CSP – CSP-Prover in Practice. In H.-D. Haasis, H.-J. Kreowski, and B. Scholz-Reiter, editors, *LDIC 2007*, pages 425–442. Springer, 2008.
- [17] Y. Isobe and M. Roggenbach. Verifying the uniform candy distribution puzzle with CSP-Prover. In S. Gruner and B. Watson, editors, *From Operations Research to Software Engineering and Beyond – Festschrift at the the Occasion of the 60ties Birthday of Derrick Kourrie*. Technical Report, University of Pretoria, 2008.
- [18] Y. Isobe, M. Roggenbach, and S. Gruner. Extending CSP-Prover by deadlock-analysis: Towards the verification of systolic arrays. In *FOSE 2005*, Japanese Lecture Notes Series 31. Kindai-kagaku-sha, 2005.
- [19] F. S. E. Limited. Failures-divergence refinement: FDR2. <http://www.fs.el.com/>.
- [20] T. Nipkow, L. C. Paulon, and M. Wenzel. *Isabelle/HOL*. Springer, 2002.
- [21] L. O'Reilly, Y. Isobe, and M. Roggenbach. Integrating Theorem Proving for Processes and Data. In M. Haverdeen, J. Power, and M. Seisenberger, editors, *CALCO-jnr 2007*. University of Bergen, 2008.
- [22] A. Roscoe. *The theory and practice of concurrency*. Prentice Hall, 1998.
- [23] P. Ryan, S. Schneider, M. Goldsmith, G. Lowe, and B. Roscoe. *The Modelling and Analysis of Security Protocols: the CSP Approach*. Addison-Wesley, 2001.
- [24] N. Shankar. PVS: Combining specification, proof checking, and model checking. In M. K. Srivas and A. J. Camilleri, editors, *Formal Methods in Computer-Aided Design*, LNCS 1196, pages 257–264. Springer, 1996.
- [25] H. Tej and B. Wolff. A corrected failure-divergence model for CSP in Isabelle/HOL. In J. Fitzgerald, C. B. Jones, and P. Lucas, editors, *FME'97*, LNCS 1313, pages 318–337. Springer, 1997.
- [26] K. Wei and J. Heather. Embedding the stable failures model of CSP in PVS. In J. Romijn, G. Smith, and J. van de Pol, editors, *Integrated Formal Methods*, LNCS 3771, pages 246–265. Springer, 2005.