Fond (and Frank) Memories of Frank

Prakash Panangaden(✉)

School of Computer Science, McGill University, 3480 Rue University, Room 318, Montréal, QC H3A 0E9, Canada
prakash@cs.mcgill.ca

How can Frank be 60? I'm only, oh yeah wait, it all comes back to me now! Frank is so youthful and vigorous that it seems hard to believe that he is approaching 60; the new 40 as the popular saying goes. I met Frank long ago at some now forgotten conference. However, it was in 1990 that I spent a week in Amsterdam visiting the CWI and again in 1992, when I spent a month at CWI that I really got to know him well. We were friendly rivals over concurrent constraint programming and friends on all topics in semantics of concurrency. Of course, we were bitter enemies on the squash court where I consistently thrashed him, but perhaps he remembers it differently! Sadly in the late 1990s our interests diverged when I pursued probabilistic systems and Frank continued with different directions in concurrency theory.

Let me describe our work in concurrent constraint programming. In the late 1980's there was much excitement about the "Japanese Fifth Generation Project" which aimed to build massively parallel machines that were geared towards symbolic computing rather than numerical computation. The language of choice was concurrent logic programming of which there were several variations. Some order was brought to this world by Vijay Saraswat with his pioneering thesis on Concurrent Constraint Programming [Sar87,Sar90]. This was quickly followed by denotational semantics for this family of languages developed in [SRP91] and independently in [dBP90]. The work in [SRP91] built on ideas from [JPP89] whereas the ideas in [dBP90] were a remarkable precursor to the later proliferation of game semantics.

What was striking about concurrent constraint programming was that it allowed "side effects" but in a graceful way. One could update data structures by adding information, but one could not take away information as with the brutal assignment statement of imperative programming languages. The conceptual model was as follows. There is a repository of "information" called the store. The store is just a first-order formula in some logical language. This logic is equipped with a notion of entailment; how entailment queries are answered is abstracted away. Several processes are allowed to run concurrently and interact with the store by either adding information to the store, a so-called **tell** operation or they can **ask** whether a formula is entailed by the store. Syntactically one can write `tell(f)` and `ask(f) ---> P`. The former just adds the formula f to the store and the latter asks whether the store entails the formula f. If it does the process continues by executing P, otherwise it suspends. Note that this is not an if-then-else; an **ask** never returns false. A suspended process may wake up later on if and when some other process adds information to the store which makes

E. Ábrahám et al. (Eds.): de Boer Festschrift, LNCS 9660, pp. 6–8, 2016.
DOI: 10.1007/978-3-319-30734-3_2

it strong enough to imply the guard. Thus the "ask" is really a synchronization mechanism.

The insight of de Boer and Palamidessi was to model this as a dialogue between processes. Each process was viewed as a sequence alternately posing questions and answering other questions. They imposed clever closure conditions on these interaction sequences and came up with a fully abstract model for concurrent constraint programming. Essentially the same ideas appear in the Saraswat *et al.* paper but with the added twist that the interactions were modelled with closure operators so the way that the closure conditions were expressed were different. Later it was realized that this kind of model has exactly the same algebra as the existential fragment of first-order logic [MPSS95]: ask was a weak form of implication, parallel composition was "and" and block structuring was existential quantification. Concurrent constraint programming evolved many interesting variations later involving time [SJG94,SJG96], continuous change [GJS98] probability [GJS97,GJP99] and so forth. Indeed there has been recent work on adding epistemic modalities to the logic [CKP09].

It was an exciting time for concurrency and I am happy to say concurrency theory continues as vibrant as ever and Frank continues to do pioneering work in the area. So here's to you Frank, many happy returns of the day.

References

[CKP09] Chatzikokolakis, K., Knight, S., Panangaden, P.: Epistemic strategies and games on concurrent processes. In: Nielsen, M., Kučera, A., Miltersen, P.B., Palamidessi, C., Tůma, P., Valencia, F. (eds.) SOFSEM 2009. LNCS, vol. 5404, pp. 153–166. Springer, Heidelberg (2009)

[dBP90] de Boer, F.S., Palamidessi, C.: On the asynchronous nature of communication in concurrent logic languages: a fully abstract model based on sequences. In: Baeten, J.C.M., Klop, J.W. (eds.) CONCUR 1990. LNCS, vol. 458, pp. 99–114. Springer, Heidelberg (1990)

[GJP99] Gupta, V., Jagadeesan, R., Panangaden, P.: Stochastic processes as concurrent constraint programs. In: Proceedings of the 26th Proceedings Of The Annual ACM Symposium On Principles Of Programming Languages, pp. 189–202 (1999)

[GJS97] Gupta, V., Jagadeesan, R., Saraswat, V.: Probabilistic concurrent constraint programming. In: Mazurkiewicz, A., Winkowski, J. (eds.) CONCUR 1997. LNCS, vol. 1243, pp. 243–257. Springer, Heidelberg (1997)

[GJS98] Gupta, V., Jagadeesan, R., Saraswat, V.A.: Computing with continuous change. Sci. Comput. Program. **30**(1), 3–49 (1998)

[JPP89] Jagadeesan, R., Panangaden, P., Pingali, K.: A fully abstract semantics for a functional language with logic variables. In: Proceedings of IEEE Symposium on Logic in Computer Science, pp. 294–303 (1989)

[MPSS95] Mendler, N.P., Panangaden, P., Scott, P.J., Seely, R.A.G.: A logical view of concurrent constraint programming. Nord. J. Comput. **2**, 182–221 (1995)

[Sar87] Saraswat, V.A.: The concurrent logic programming language cp: definition and operational semantics. In Proceedings of the SIGACT-SIGPLAN Symposium on Principles of Programming Languages, pp. 49–62. ACM, January 1987

[Sar90] Saraswat, V.A.: Concurrent Constraint Programming Languages. Doctoral Dissertation Award and Logic Programming Series. MIT Press, Cambridge (1990)

[SJG94] Saraswat, V.A., Jagadeesan, R., Gupta, V.: Foundations of timed concurrent constraint programming. In: Proceedings of the Ninth Annual IEEE Symposium On Logic In Computer Science, Paris, 1994, pp. 71–80. IEEE Press (1994)

[SJG96] Saraswat, V., Jagadeesan, R., Gupta, V.: Timed default concurrent constraint programming. J. Symbolic Comput. **22**(5), 475–520 (1996)

[SRP91] Saraswat, V.A., Rinard, M., Panangaden, P.: Semantic foundations of concurrent constraint programming. In: Proceedings of the Eighteenth Annual ACM Symposium on Principles of Programming Languages (1991)

http://www.springer.com/978-3-319-30733-6

Theory and Practice of Formal Methods
Essays Dedicated to Frank de Boer on the Occasion of His 60th Birthday

Ábrahám, E.; Bonsangue, M.; Johnsen, E.B. (Eds.)

2016, XII, 427 p. 112 illus. in color., Softcover
ISBN: 978-3-319-30733-6