Role model for young researchers

Markus Roggenbach visited the UP's Computer Science Department

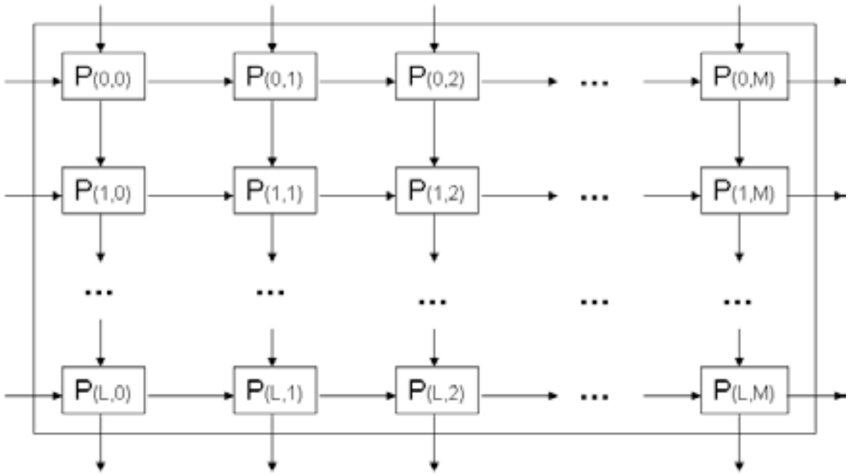
In the framework of a prestigious bi-national (British/South African) research cooperation co-funded by the British Royal-Society and the South African National Research Foundation (NRF), Markus Roggenbach followed an invitation by Derrick Kourie and Stefan Gruner and visited UP's Computer Science Department during two weeks in July 2007. This article first introduces Markus Roggenbach personally as an inspiring example and role model for the young generation of novice researchers. Thereafter, some background information about the innovative research in the context of this bi-national cooperation is described.

Markus Roggenbach is both a well-known scientist as well as a formidable artist, and the path which he would eventually choose was not easy to be foreseen. Already during his school time in Wolfsburg (Germany) he won a prize in the national music youth competition "Jugend musiziert": his favourite instrument is the oboe, but he also plays the piano very well. After his Matric (in Germany: "Abitur"), Markus studied Music at the renowned Higher Institute for Music in Detmold. After a while, a difficult decision was to be made: to become – or not to become – a professional musician: Music or Informatics? Markus decided to start with Informatics and Mathematics at the Technical University of Braunschweig, and continued his studies of Informatics at the University of Karlsruhe, where he also received his MSc-Diploma in Infomatics, with an MSc thesis on Parallel Computation and Systolic Arrays. Thereafter he moved from the University of Karlsruhe to the University of Mannheim, where he achieved his Doctorate on Abstract Characterizations of Bi-Simulation. From 1998 to 2003, Markus worked as a Post-Doc Research Fellow at the Faculty of Informatics of the University of Bremen, where his focus of research was Process Algebras and Mathematical Specifications of Software Systems. In the year 2003 he moved from Germany to Wales (GB), where he teaches mainly Formal Methods and Formal Semantics, but also Algorithms and Data Structures, at the University of Wales in Swansea. As a passionate and dedicated teacher in higher education, one of Markus' main concerns is the lack of entry-skills and intellectual maturity, thus a general "non-study-ability" amongst many students in the post-1970s mass-universities. In contrast to other recently emerging opinions, he is also a strong believer in the "mathematicalness" of Informatics (respectively the Computer Sciences). Markus has appeared on numerous international conferences, is a programme committee member to some of them, and published articles in scientific journals. He currently supervises four doctoral students: three of them are full-time (internal) students, whereas one of them is a part-time (external) student. Moreover, Markus maintains standing cooperations with the University of Bremen on Formal Semantics, with Tsukuba in Japan on Automated Theorem Proving, with the Humboldt-University of Berlin on Formal Methods of Software Testing, and now also with the University of Pretoria on CSP specifications and correctness proofs. He presented a course in South Africa in 2000, when he presented a course on Formal Methods and Algebraic Specifications as an Invited Lecturer at the University of Cape Town. Besides his scientific activities, Markus is still an active amateur musician, playing his instrument in various local trios, quartets and classical chamber orchestras, of which also some recordings exist.

→ Markus Roggenbach in Pretoria: July 2007

Background of the project cooperation

Parallel and Distributed Algorithms are very useful tools because they can exploit the combined computational power of a network or grid of processors, rather than the single processor of classical von-Neumann computer architectures. On the other hand, parallel and distributed algorithms are intrinsically difficult to design and to test, because the "path" or "trace" of a parallel computation, even based on the same set of input data, can vary from time to time and is not easy to predict. In such a setting the danger of deadlock is looming, which is a situation in which no processor can make any computational progress any more because of waiting for data from other processors which are also stuck. This is the well-known circular-wait situation, also studied in the field of operating systems. Already in 1987, Roscoe and Dathi [1] have researched the circumstances under which a distributed system can be regarded as *deadlock-free*, in other words: what must be the case such that we can necessarily conclude that the above-mentioned circular-wait situation is impossible? According to [1], a progress-function $f(T,P)$ for traces *T* and processors *P* must be discovered such that (basically, in a somewhat simplified explanation): for each two processors *Pi* and *Pj* and for each of their traces *Ti* and *Tj* the following condition holds: whenever there is an un-granted communication request from *Pi* to *Pj*, then $f(Ti,Pi) > f(Tj,Pj)$. If such a progress-function can be found, then the distributed algorithm is guaranteed to be deadlock-free. The problem in this context is two-fold: (a) such a function, which differs from algorithm to algorithm, is hard to be discovered for a particular algorithm under investigation, and (b) when it is discovered then its discovery is mostly based on the exploration of only a small numbers of sample processors within the distributed computation network – not the entire network as a whole. Therefore the question arises: whenever we have "guessed" (based on a small number of observation examples) a candidate function f, how can we be sure that this f is actually valid for the entire network and not only for the small number of example processors which we have observed? This is exactly the point where the availability of automated theorem proving cannot be valued highly enough, and this is also the context of the above-mentioned bi-national (in fact even multi-national) project cooperation. Based on the general theorem prover tool "Isabelle" [2], Isobe has programmed a "plug-in" which is suited for proofs about parallel and distributed algorithms described in the network description language CSP [3]. With this tool, for example, the deadlock-freeness of a particular systolic array described by Kung for the purpose of matrix multiplication [4] was formally (i.e. automatically) proven for the very first time two years ago [5]. A systolic array is a highly symmetric (usually rectangular) grid of fully synchronised parallel-processing in which the data are "pumped" through the network in "wave-fronts" following the "heart-beat" of a global clock.



→ *1. Sketch of a systolic array for matrix multiplication*

This innovation in automated deadlock analysis (which could only be dreamt about 20 years ago when Roscoe and Dathi first published their mathematical methodology) was made possible by the advances in both software engineering and computer hardware development. The above-mentioned project cooperation aims at widening and deepening our knowledge in this domain by proving the deadlock-freeness of a larger collection of relevant parallel or distributed algorithms and thereby also refining the "tricks" and techniques of the tool-supported proof-methodology itself.

In this most recent visit to South Africa (Cape Town, November 2008), Markus presented his work as an invited keynote speaker at the prestigious IEEE conference on Software Engineering and Formal Methods [6].

Prospective MSc/PhD students, who are ambitious enough for the challenges of this truly difficult subject, are encouraged to contact Dr Stefan Gruner (sg@cs.up.ac.za) at the University of Pretoria for further information.

Literature References:

[1] A. Roscoe and N. Dathi: The Pursuit of Deadlock Freedom. Information and Computation Vol.75, No.3, pp.289-327, 1987.

[2] T. Nipkow, L. Paulon and M. Wenzel: Isabelle/HOL. Lecture Notes in Computer Science Vol.2283, Springer-Verlag, 2002.

[3] Y. Isobe and M. Roggenbach: A Generic Theorem Prover of CSP Refinement. Proceedings TACAS'05, Lecture Notes in Computer Science Vol.3440, pp.108-123, Springer-Verlag, 2005.

[4] H. Kung: The Structure of Parallel Algorithms. Advances in Computers, Academic Press, 1980.

[5] Y. Isobe, M. Roggenbach and S. Gruner: Extending CSP-Prover by Deadlock-Analysis towards the Verification of Systolic Arrays. Proceedings FOSE'05: 12th Japanese Workshop on the Foundations of Software Engineering, 2005.

[6] M. Roggenbach: Tools for CSP. Invited keynote lecture. In A. Cerone and S. Gruner (eds.): 6th IEEE International Conference on Software Engineering and Formal Methods, Cape Town, Nov. 2008, IEEE Computer Society Press.