Rajshekar K

Department of Computer Science and Engineering,
Indian Institute of Technology Dharwad (IITDh),
Dharwad, Karnataka - 580011,
India

☎ (+91) 836 2212 888
✉ rajshekar.k@iitdh.ac.in
www.cse.iitd.ac.in/∼rajshekar

Personal Profile

I am currently an Assistant Professor in the Dept. of Computer Science and Engineering, IIT Dharwad. I have completed my PhD from IIT Delhi, and served as a Senior Research Fellow in the same department. My research areas include computer architecture, hardware reliability, hardware security and accountability, and architectural simulation.

Education

2012 - December 2016	Ph.D. in Computer Science and Engineering *Indian Institute of Technology Delhi*, India Thesis: **Employing Redundancy based Techniques to Provide Reliability, Security and Accountability in Modern Processors**	
2010 - 2012	M.Tech. in Computer Science and Engineering *Indian Institute of Technology Delhi*, India Thesis: **A Novel Configuration of Checker Architectures**	9.3/10 (CGPA)
2006 - 2010	B.E. in Information Science and Engineering *B.M.S. College of Engineering*, Bangalore, India *Visvesvaraya Technological University*, Karnataka, India	83.60%

Research Work

✠ Reliability against Soft Errors

We proposed a novel, assisted re-execution based technique, *FluidCheck* [J3], to efficiently detect the occurrence of soft errors in many-core processors. The primary contribution is the proposed processor architecture that firstly, allows dynamic and flexible migration of both the primary and the speculative threads, and secondly, reduces the pressure of execution assistance on the NoC. The secondary contribution is a family of scheduling heuristics that govern the dynamic migration of threads based on the behavior of the multiprogrammed workload running on the processor. We demonstrated that 16-core processors can be protected against soft errors with a mere 27% loss in performance, surpassing traditional schemes by 42%.

✠ Security against Hardware Trojans

We proposed a novel Hardware Trojan detection scheme, *SecCheck* [C3], for systems containing ICs designed and/or fabricated by untrusted parties. *SecCheck* advocates employing slow, trusted, *home-grown* cores to verify the functioning of fast, untrusted, *third-party* cores. The verification is done by adapting traditional reliability techniques. The overhead of verification is reduced by exploiting opportunities of temporally overlapping computation and verification. We demonstrated that the technique can ideally provide security against Hardware Trojans with a mere 10-17% penalty in performance. We additionally proposed heuristics to quickly compute an efficient schedule of the applications, modeled as task graphs, on the processor cores. The heuristics improve the speed of scheduling 500 times over the ILP-based optimal scheduler, with a $< 1\%$ degradation in schedule quality.

✠ Accountability in 3PIP-containing SoCs

Modern SoCs contain components designed by multiple organizations, and this diversity is expected to increase. If a chip miscomputes or underperforms, fairly evaluating the roles that different components had to play is not a straightforward issue. We recognized this problem of accountability in the realm of heterogeneous SoCs. Debugging SoCs requires run-time information to help debug engineers understand and re-create the bug. Such information collection should be fair, and should not be the responsibility of a single party. This would allow for potential tampering of logs to deflect the blame for the bug. We proposed that fair information collection, required for accountability, can be achieved if every message between components from different organizations is audited fairly [J1]. As part of this work, we proposed a novel model of representing and classifying SoCs, and for the different SoC classes, we presented a range of auditing solutions that are constructed using redundancy and cryptographic principles. We presented a novel game-theoretic approach towards proving (or disproving) that a particular auditing solution is feasible or *resilient* against any attempt by the parties to sabotage the audit. We use this methodology to prove the resilience of our proposed solutions. We have also presented a thorough design and evaluation of one particular auditing solution geared for a class of SoCs that we believe to be the most prevalent. We showed that complete and fair accountability can be achieved with a mere 0.49% performance and 0.194% area overhead.

✠ Hardware Implementation of the kCAS Primitive

Modern processors provide atomic instructions such as compare-and-set (CAS) that work on exactly one address. The implementation of lock-free data structures can be greatly simplified if the hardware provides a *kCAS* instruction – an instruction that performs the compare-and-set on k addresses. Along with a Master's student, I proposed a design to achieve the kCAS instruction [C2]. We believe we are the first to attempt this. The design is essentially an extension of the coherence protocol, with additional memory per core to maintain the state of an ongoing kCAS instruction. We also proposed an optimization that performs an early back-off when we can determine that the ongoing kCAS is going to fail. This reduces the time spent on kCASes and hence improves performance. The kCAS instruction not only enables the programmer to implement and debug parallel data structures easily, but also improves the performance of the application by 4X on average. Our design has a minimal area overhead of 0.0016%.

✠ Tejas Architectural Simulator

Our research group has developed an architectural simulator, *Tejas* [C5], that is capable of simulating state-of-the-art multi-core processors, GPUs and SoCs. It has been validated against real hardware, and demonstrated to be faster than other cycle accurate simulators. *Tejas* has been released under the open source Apache v2 license. It is currently used in 51 countries by over 950 users, for both teaching and research purposes. I am among the chief designers, developers and maintainers of *Tejas*. Please visit the *Tejas* web page for more information.

✠ Techniques to Improve the Performance of an Architectural Simulator

One way to accelerate the simulation of parallel benchmarks is to have multiple simulator threads, each simulating a different core (or set of cores) of the simulated architecture [J2]. There are two issues with this approach however. Firstly, accesses to shared structures like the last level cache or the directory requires synchronization between the threads. We proposed a novel lock-free implementation of the ports that govern the access to the shared structures.

This helped reduce the penalty of synchronization to a large extent. Secondly, the individual simulator threads progress at different rates. Thus, the simulator threads need to be periodically synchronized to keep the clock skew within acceptable bounds. Instead of a regular barrier, we proposed to use an advanced variant of a barrier known as a phaser for this purpose. Our techniques helped us achieve a performance gain of around 11X when employing 64 simulator threads, while limiting the error to 2-4%.

We proposed another technique to improve simulator performance, this time for serial benchmarks. The idea is based on the exploitation of the following fact: the architectural state at a certain point in the simulation, say at the x^{th} instruction, can be approximately attained by simulating only the last δ instructions before the x^{th} one. Every instruction, right from the first to the x^{th}, need not be simulated. With this principle, the simulation can be broken down into disjoint, contiguous chunks of instructions. Each chunk can be simulated in parallel, and the results can be aggregated when all the chunks are simulated. We demonstrate that this technique can obtain speedups of up to 10X when a 16-way chunking is performed.

Publications

Journals

J1. Providing Accountability in Heterogenous Systems-on-Chip by **Rajshekar Kalayappan**, Smruti R. Sarangi, ACM Transactions on Embedded Computing Systems (TECS) (accepted for publication)

J2. ParTejas : A Parallel Simulator for Multicore Processors by Geetika Malhotra, **Rajshekar Kalayappan**, Seep Goel, Pooja Aggarwal, Abhishek Sagar, Smruti R. Sarangi, ACM Transactions on Modeling and Computer Simulation (TOMACS). Volume 27, Issue 3, September 2017

J3. FluidCheck: A Redundant Threading-Based Approach for Reliable Execution in Manycore Processors by **Rajshekar Kalayappan**, Smruti R. Sarangi, ACM Transactions on Architecture and Code Optimization (TACO). Volume 12 Issue 4, January 2016
Presented at European Network on High Performance and Embedded Architecture and Compilation Conference (HiPEAC'16), Prague, Czech Republic, 2016

J4. Surveillance using non-stealthy sensors: A new intruder model by Amitabha Bagchi, **Rajshekar Kalayappan**, Surabhi Sankhla, Wiley Security and Communication Networks. Volume 7, Issue 11, November 2014

J5. A survey of checker architectures by **Rajshekar Kalayappan**, Smruti R. Sarangi, ACM Computing Surveys (CSUR). Volume 45, Issue 4, August 2013

Conferences

C1. A Hardware Implementation of the kCAS Synchronization Primitive by Srishty Patel, **Rajshekar Kalayappan**, Ishani Mahajan, Smruti R. Sarangi, Design, Automation and Test in Europe (DATE'17), Lausanne, Switzerland, 2017.

C2. SecCheck : A Trustworthy System with Untrusted Components by **Rajshekar Kalayappan**, Smruti R. Sarangi, IEEE Computer Society Annual Symposium on VLSI (ISVLSI'16), Pittsburgh, USA, 2016.

C3. SecX: A Framework for Collecting Runtime Statistics for SoCs with Multiple Accelerators by **Rajshekar Kalayappan**, Smruti R. Sarangi, IEEE Computer Society Annual Symposium on VLSI (ISVLSI'15), Montpellier, France, 2015

C4. Tejas: A java based versatile micro-architectural simulator by Smruti R. Sarangi, **Rajshekar Kalayappan**, Prathmesh Kallurkar, Seep Goel, Eldhose Peter, IEEE International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS'15), Salvador, Brazil, 2015

Patents

P1. Parallelization of a Trace-Driven Architectural Simulator, by Smruti R. Sarangi, **Rajshekar Kalayappan**, Avantika Chhabra (under process)

P2. Providing Accountability in Heterogeneous SoCs Using Dual Metering, by Smruti R. Sarangi, **Rajshekar Kalayappan** (under process)

Professional Service and Activities

✠ Reviewer

- Journals: TCAD, CSUR
- Conferences: MICRO, ASPLOS, IPDPS, HiPC, VLSI
- Funding proposals submitted to Science and Engineering Research Board, Government of India

✠ Teaching Assistant

- UG courses: Basic Computer Architecture, Digital Hardware Design, Introduction to Computer Science, Data Structures and Algorithms
- PG courses: Architecture of High Performance Computers (Outstanding TA Award)

References

Dr. Smruti R. Sarangi
Department of Computer Science and Engg.
Indian Institute of Technology Delhi
Hauz Khas, New Delhi - 110 016, India
☎ +91 11 2659 7065
✉ srsarangi@cse.iitd.ac.in

Prof. Preeti Ranjan Panda
Department of Computer Science and Engg.
Indian Institute of Technology Delhi
Hauz Khas, New Delhi - 110 016, India
☎ +91 11 2659 6030
✉ panda@cse.iitd.ac.in

Prof. M Balakrishnan
(Deputy Director (Strategy & Planning) IIT Delhi)
Department of Computer Science and Engg.
Indian Institute of Technology Delhi
Hauz Khas, New Delhi - 110 016, India
☎ +91 11 2659 1285
✉ mbala@cse.iitd.ac.in

Dr. Osman Unsal
Computer Architecture For Parallel Paradigms
Group Manager
Barcelona Supercomputing Center
Nexus I - Planta 3 C/ Gran Capita, 2-4
Barcelona 08034. Spain.
☎ +34 934054293
✉ osman.unsal@bsc.es

Prof. Anshul Kumar
Department of Computer Science and Engg.
Indian Institute of Technology Delhi
Hauz Khas, New Delhi - 110 016, India
☎ +91 11 2659 1321
✉ anshul@cse.iitd.ac.in

Dr. Kolin Paul
Department of Computer Science and Engg.
Indian Institute of Technology Delhi
Hauz Khas, New Delhi - 110 016, India
☎ +91 11 2659 6033
✉ kolin@cse.iitd.ac.in