Radu Grigore

November 2009

Mail: UCD CASL, University College Dublin, Belfield, Dublin 4, Ireland
Email: radugrigore@gmail.com
Homepage: http://rgrig.appspot.com/
Blog: http://rgrig.blogspot.com/

Education

2005–2010 PhD at UCD (University College Dublin) on *The Design and Algorithms of a Verification Condition Generator*, funded by the European research project MOBIUS.

1998–2003 BSc at PUB (Politehnica University of Bucharest). My average grade is 9.75 out of 10. My diploma dissertation is on *Traffic Models for Data Networks*.

2001 Internship in the Darmstadt Institute for Microelectronics, funded by a DAAD scholarship.

1994–1998 "Gheorghe Șincai" highschool, București

Industry

2008 I experimented with using kd-trees and R-trees to add orthogonal range queries on top of BigTable as a Software Engineer Intern in Google.

2004–2005 I helped design and develop a translator between two languages used in hardware verification. The client was Synopsys and I was working in a team of ten within NoBug Consulting.

2003–2004 I designed and developed a PSL frontend for the model checker RuleBase while working for NoBug Consulting.

2003 I helped design and develop a translator from PSL to finite automata with counters, whose implementation is generated in a variety of procedural languages. I worked in a team of two within NoBug Consulting.

1999 I maintained the Neuron Geographical Information System.

Teaching

2009 *Unix Programming* (UCD): I designed the course and I delivered the lectures.

2007–2008 Coach of the team that represented UCD (and Ireland) in ACM ICPC.

2008 *Foundations of Computing* (UCD): I demonstrated during practicals.

2008 *Operating Systems* (UCD): I designed the course and I delivered the lectures.

2006–2007 *Algorithmic Problem Solving* (UCD): I demonstrated during practicals.

2006 *Data Structures and Algorithms* (UCD): I demonstrated during practicals.

2004 *Telecommunications Software* (PUB): I demonstrated during practicals.

2003 *Data Networks* (PUB): I demonstrated during practicals.

Research Publications

2010 Mikoláš Janota, Goetz Boetterweck, Radu Grigore, and Joao Marques-Silva. *How to Complete a Configuration Process?* Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)

2009 Radu Grigore, Julien Charles, Fintan Fairmichael, and Joseph Kiniry. *Structured Postcondition of Unstructured Programs*. ECOOP Workshop on Formal Techniques for Java-like Programs (FTfJP)

2009 Mikoláš Janota, Fintan Fairmichael, Viliam Holub, Radu Grigore, Julien Charles, Dermot Cochran, and Joseph Kiniry. *CLOPS: A DSL for Command Line Options*. IFIP Working Conference on Domain Specific Languages (DSL WC)

2007 Radu Grigore and Michał Moskal. *Edit and Verify*. Workshop on First-Order Theorem Proving (FTP)

2007 Mikoláš Janota, Radu Grigore, and Michał Moskal. *Reachability Analysis for Annotated Code*. ESEC/FSE Workshop on Specification and Verification of Component-Based Systems (SAVCBS)

2005 Valentin Ștefan Gheorghiță and Radu Grigore. *Constructing Checkers from PSL Properties*. Conference on Control Systems and Computer Science (CSCS)

Open Source Contributions

2007... FreeBoogie is a program verifier backend.

2007... AstGen generates code from abstract grammars. It is part of FreeBoogie.

2008... CLOPS is a code generator for command line parsing.

2005–2009 I maintained ESC/Java and I implemented its reachability analysis.

2008 Part of my work in Google is included in UzayGezen.

2005 CFind indexes and searches a hard-drive. I might improve it sometime.

Personal Statement

I want to build bridges in computing science. The largest rift is between theoreticians and practitioners, but it is certainly not the only one. The research community is partitioned into small enclaves, each with its own language and its own norms. Such grouping is desirable as it is impossible for one person to understand all the recent research results in computing science. Individuals cope with information overload by spending more time to digest information from trusted and familiar sources than from other sources. In such an environment it is not too surprising that sometimes surprising links are found between the results of different communities. I want to spend most of my time searching for connections between the results of various communities and translating from one language into another. The task is hard because it involves being abreast with more than one field.

The areas I enjoy most are *program verification*, *programming languages*, and *algorithms*. The goal of the program verification community is to find good methods to reason formally about the correctness of programs; the goal of the programming language community is to devise expressive and robust ways to capture algorithms; the goal of the algorithms community is to find efficient ways to solve repetitive problems. I want to continue searching for links between these areas and, in the process, develop tools that practitioners can use to write better software.