CV

Alexey Bakhirkin

Email	abakhirkin@gmail.com
Occupation	PhD student in Computer Science, University of Leicester
Now living in	Leicester, UK
Citizenship	Russian
Born	1987 in Moscow, Russia

Field of interest Formal methods and verification

About me I'm a PhD student in University of Leicester, UK, supervised by *Nir Piterman*. I'm in the final 4th year of studies (writing-up stage in UK's formal terms), and will submit my thesis in *May 2016*. My research is into *program analysis*, and specifically – into proving non-termination of programs using abstract interpretation. So far this work has led to three publications [2, 3, 4].

Before that, while studying for an Engineering degree (similar to MSc) at Bauman University in Moscow, I got interested in model checking and concurrent programming, which led to a publication on software transactional memory [1] (in Russian).

For about five years (2007–2012) I was working as a programmer in Moscow. Most of my experience is with .NET and Java; I'm also familiar with C, C++, Python, Scala (in a sense that I can write code in those, but never participated in a big project).

What I'm looking for I'm looking for a position to do research in the field of formal methods and verification. My current background is in program analysis and abstract interpretation, but I know a few things about model checking, shape analysis, logic – and I will be happy to work in a field related to those. I'd say that I have strong programming skills (incl. professional programming experience), and I will be happy to work both on theory and implementations.

Brief Timeline

This is a very brief timeline. The details (what exactly I was doing) are below.

Jun 2012 – present	PhD Student in Computer Science, University of Leicester, UK. Supervisor: *Nir Piterman*
Aug – Oct 2014	Intern at Microsoft Research, Cambridge, UK. Host: *Josh Berdine*
2007 – 2012	Software Developer in Moscow, Russia
2004 – 2010	Student in Computing (Engineering degree, similar to MSc) at Bauman Moscow State University, Russia

Publications

[1] Alexey Bakhirkin. A comparison of blocking and non-blocking synchronization in object-based software transactional memory. In *PACO*, 2010. In Russian, available here.

[2] Alexey Bakhirkin, Josh Berdine, and Nir Piterman. Backward analysis via over-approximate abstraction and under-approximate subtraction. In *SAS*, 2014. Available here.

[3] Alexey Bakhirkin, Josh Berdine, and Nir Piterman. A forward analysis for recurrent sets. In *SAS*, 2015. Available here.

[4] Alexey Bakhirkin and Nir Piterman. Finding recurrent sets with backwards analysis and trace partitioning. In *TACAS*, 2016. To appear, available here.

Talks

Finding Recurrent Sets with Backward Analysis and Trace Partitioning – paper presentation as TACAS 2016. Slides – here.

A Forward Analysis for Recurrent Sets – paper presentation at SAS 2015 conference. Slides – here.

Does My Program Ever Finish – a brief introduction to (non-)termination. At BCS event at Leicester, 2015. Slides – here.

Backward Analysis via Over-Approximate Abstraction and Under-Approximate Subtraction – paper presentation as SAS 2014. Slides – here.

Software Transactional Memory – at Moscow ALT.NET group meeting, 2011. In Russian. Video – here.

Other

I was a teaching assistant for several modules (obviously, not all at the same time): C++ programming, model checking (MSc level), Java programming, Haskell programming, operating systems (BSc level). I've supervised a group of BSc students on a small project involving researching a topic and preparing a report and talks on it.

I was organizing the departmental PhD seminars – a regular event where we invite PhD student from British universities to talk about their research.

In August 2013, I attended Marktoberdorf Summer School on Software Systems Safety.

Languages English – fluent, Russian – native.

Links University web page – here. DBLP – here

Detailed timeline

June 2012 – present PhD Student in Computer Science, University of Leicester, UK. Supervisor: *Nir Piterman*. My current research is into extending abstract interpretation techniques to allow proving non-termination of programs. Specifically, we're more interested programs that require complicated abstract domains to analyse them (e.g., programs that use dynamic memory and thus need to be analysed using shape analysis techniques). Our first step in this direction is [3] which is based on forward analysis. Another our work [4] uses a combination of more complicated techniques (backward analysis, trace partitioning) to achieve better results for numeric programs. Overall, I think that extending tools and techniques for safety, in a way that allows more-than-safety reasoning (e.g., reasoning about (non-)termination) is both an interesting and useful direction.

The initial direction of my PhD was to improve safety checking of heap-manipulating programs (those that use dynamic memory). After a number of experiments with TVLA (an interesting shape-analysis framework based on 3-valued logic) we got to a conclusion that sophisticated safety reasoning can (and perhaps, should) benefit from the results inferred by termination or non-termination provers. Paper [2] is built around this observation.

Aug – Oct 2014 Intern at Microsoft Research, Cambridge, UK. Host: *Josh Berdine*. The goal of the project was to better understand scheduling in abstract interpretation. Abstract interpretation can be seen as iteratively solving a system of dataflow equations (where every program statement contributes an equation). The order in which we iterate over the equations matters in practice, and we tried to formalize some knowledge and experience that was accumulated in MSR on this topic. During the internship, I learned a few things about the more complicated parts of abstract interpretation: about dealing with control flow graphs, about inter-procedural analysis, and etc.

2011 – 2012 Research in Bauman University. While contemplating the option of going into a PhD, I kept in touch with my supervisor from Bauman university and was preparing for possibly starting a PhD there. The project that we had in mind was about model checking of reactive programs: that involve event dispatch routines, make use of asynchrony and callbacks. This could be useful, e.g., to verify implementations of network protocols or device drivers. In part-time mode, I was doing what 1st-year PhD students usually do: reading papers, learning algorithm and tools. I learned about temporal logics, got my hands on SPIN and TLA+ (apparently, it is now used at Amazon).

2007 – 2012 Software Developer in Moscow, Russia. I was focusing on C# and .NET, also had experience with Oracle database and some Oracle middleware. I did work for a couple of smaller companies, but also for two bigger IT corps: CROC (2011–2012) and LANIT (2008 – 2011). Both implement interesting and complex projects for large businesses and government agencies.

Sep 2004 – Jul 2010 Student in Computing (Engineering degree, similar to MSc) at Bauman Moscow State University, Russia. My final project was on software transactional memory – I implemented a couple of STM algorithms (in C#) and performed some performance measurements. The project eventually led to a publication [1]. This was my real introduction to concurrency, memory models, and non-blocking synchronization.