Antrag auf Gewährung einer Unterstützung zum Aufbau internationaler Kooperationen

Grant Application to Support the Initiation of International Collaboration

Gerald Lüttgen (U. Bamberg, Germany) and Richard Paige (U. York, U.K.)

1 General Information

This is an application for a grant to support the initiation of international collaboration.

1.1 Applicant

Surname, given name, title	Lüttgen, Gerald, Prof. Dr.
Position	Universitätsprofessor
A previous DFG grant no.	LU 1748/1-1
Affiliation	Lehrstuhl für Softwaretechnik und Programmiersprachen, Fakultät für Wirtschaftsinformatik und Angewandte Informatik, Otto-Friedrich-Universität Bamberg
Work address	Wilhelmsplatz 3, 96047 Bamberg, Germany
Phone / Fax	+49 (0)951 863-3850 / +49 (0)951 863-3855
Email	gerald.luettgen@swt-bamberg.de

Surname, given name, title	Paige, Richard, Prof.
Position	Professor (Personal Chair)
A previous DFG grant no.	None
Affiliation	Enterprise Systems Research Group Department of Computer Science, University of York
Work address	Deramore Lane, York YO10 5GH, Großbritannien (U.K.)
Phone / Fax	+44 (0)1904 325-5170 / +44 (0)1904 325-5599
Email	paige@cs.york.ac.uk

Prof. Paige has recently been awarded matching funds from the University of York. These will pay for one two-day bilateral workshop at York and also for a short visit of the two senior participants from Bamberg to York, which will take place between May and July 2012.

1.2 Topic

Advanced Heap Analysis and Verification

1.3 Scientific Discipline & Field of Work

Computer Science *(409 – Informatik)*; Theoretical Computer Science *(409-01 – Theoretische Informatik)*; Software Technology *(409-02 – Softwaretechnologie)*; Formal Methods of Software Verification *(Formale Methoden der Softwareverifikation)*

1.4 Funding Type

Workshop in Germany + guest visits

1.5 Application Period

October 2012 – February 2013

1.6 Abstract

The formal verification of software is key to guaranteeing their safe operation and a grand challenge in Computer Science. A main difficulty concerns reasoning about pointers and dynamic data structures stored in the computer's heap, which are indispensable in today's software but make it hard to debug. Several techniques have been proposed to address aspects of pointer programs, heap analysis and heap verification, together with prototypic tool support. However, their practical impact is currently low since they do not interoperate, and since each technique targets only specific pointer structures and does not scale beyond small programs.

This proposal is intended to bootstrap a collaboration between internationally recognized researchers of the Universities of Bamberg, Germany, and York, U.K, which will address these shortcomings. Two bilateral workshops will give the researchers a platform to exchange their expertise and identify synergies between their approaches to heap analysis and verification. In particular, requirements for a domain-specific language for representing and reasoning about pointer programs and heap structures shall be identified, so as to enhance interoperability of approaches and tools. The workshops will incubate new ideas and, together with mutual short visits, will lead to the submission of a large-scale joint DFG/EPSRC proposal by February 2013.

2 State of the Art

We first provide an overview of heap analysis and verification, which crosses the Computer Science disciplines of Programming Languages, Software Engineering and Automated Verification.

The heap is a memory area that allows programs to dynamically allocate and deallocate memory at runtime. Practically all of today's software systems make use of the heap for dynamic memory management. Since the heap is generally unbounded and enables the storage of arbitrary data structures via pointers, it is difficult to ensure that programs operate according to expectations. An additional complication is that the heap can be used by many processes or threads in parallel. Competition between programs for heap control and programming errors involving heap pointers are significant sources of software failures. Therefore, ensuring that the heap is well formed and properly maintained is vital to building reliable software systems.

Prior research at Bamberg and York. The University of Bamberg has a number of internationally recognized researchers working in the area of this proposal, in its Software Technologies and Informatics Theory research groups (headed by Prof. Lüttgen and Prof. Mendler, respectively), as has the University of York within several research groups of its Department of Computer Science (Prof. Paige, Dr. Plump, Prof. Runciman and Prof. Woodcock).

Prior research at Bamberg's Software Technologies Group includes symbolic object code analysis [14, 15], which is based on bounded symbolic execution and path-sensitive slicing and employs an SMT solver as its execution and verification engine. Experiments conducted with thousands of Linux device drivers show that this method performs well in practice for checking pointer safety. In [22, 23], a graph-based framework is advocated that allows for the verification of properties of heap-manipulating programs such as pointer and shape safety, pointer aliasing and termination. It supports concurrency including thread creation at runtime, and its tool implementation Juggrnaut [10] has been applied successfully to list and tree algorithms.

Research in Bamberg's Informatics Theory Group focuses on contextual enrichments of constructive, intuitionistic logics and their computational semantics. These modal type theories [13] extend classical semantics of modal and description logics by intensional and contextual aspects, which makes them adequate for abstraction and refinement. In [12], it is demonstrated how a constructive multi-modal logic may serve as a semantic type system for expressing computations in contextual, tree-shaped data structures. Through modal types, contextual visibility of pointer references and properties of heap memory protection can be specified and verified.

Past research at York includes research on the Eiffel Refinement Calculus [18] and Unifying Theories of Programming [9] (both of which developed theories of heaps), research on model and graph transformation [1, 2, 8] (which provides abstractions of heaps and mechanisms for their manipulation), and more generally, research on concurrency theory and reactive systems. Concretely, the proposed project will be supported by past projects funded by the UK's Engineering and Physical Sciences Research Council (EPSRC), including: "Refinement Patterns for Contractual Statecharts" which developed tools that could be used for enabling heap verification, and "Systems Development: Domain-Specific Modelling" which focused on the verification of complex programs that employ heaps. Further support will come through technology developed in four European projects (MODELWARE, MODELPLEX, INESS and MADES), in which York participated. Especially, the Epsilon technology (www.eclipse.org/epsilon) will provide the basis of a set of software tools for automating parts of the heap verification process.

Prior research by others. Heap analysis has recently received much attention. The following gives a brief overview of selected prior research outside Bamberg and York, which typically relies on general program abstraction, modelling and verification techniques.

Dataflow analysis is a technique for statically calculating semantic information about a program using its control-flow graph. It is efficient but restricted to rather shallow properties of pointer programs, such as aliasing relations [16], points-to information [27] and pointer range analysis [26].

Shape analysis represents data structures of unbounded size by finite structures, called shape graphs. These may be formalized by three-valued logical structures [5, 24] and obtained from pointer programs by predicate abstraction, so as to yield Boolean programs that conservatively preserve program behaviour [3, 6, 19]. Graph transformations may be used to describe the evolving shapes of heap structures by abstract graphs, and to implement pointer manipulations by graph transformation rules [20]. A grammar-based approach to heap abstraction is presented in [11]; however, it only supports tree data structures.

Separation logic permits local, logical reasoning about linked structures, enabling modular correctness proofs for pointer-manipulating programs [17, 21]. Tools such as SpaceInvader [25] and SLAyer [4] are based on separation logic and focus on the verification of device drivers, while jStar is a tool for checking given pre- and postconditions of Java methods [7].

Limitations and challenges. One of the major difficulties in static heap analysis is the heap's unboundedness, which prevents the application of standard finite-state analysis and verification techniques. While a variety of approaches for representing and reasoning about heap structures exists, most of them focus on some specific setting or data structure, thus preventing their application in the general case. Many approaches are currently incomparable and incompatible with each other, and tools based on them often do not interoperate and can only be employed within their respective, restricted application domain. In addition, existing approaches do not scale well beyond small programs.

2.1 Scientific Goals & Collaboration Concept

This project shall establish a sustainable, international research collaboration in heap analysis and verification between researchers of the Universities of Bamberg, Germany, and York, U.K.

Scientific goals. The main purpose of this collaboration is to (i) join forces in addressing the aforementioned limitations of current techniques in heap analysis and verification, (ii) develop interoperability support for future toolsets to be built at Bamberg, York and elsewhere, and (iii) collect and exchange case studies to evaluate and contrast current and future heap verification techniques. These research goals are of significant interest to a large community of researchers; progress will be enhanced by greater interaction, communication and synergy among these researchers. Thus, our main goal is to establish an international research network focusing on heap verification and funded by national research councils (in the short-term) and by the European Commission (in the long-term).

This proposal asks for bootstrapping this process by initiating a cooperation between internationally recognized researchers at the Universities of Bamberg and York in the fields of formal methods, automated verification, graph and model transformations and heap analysis. The support will help us to identify significant gaps in current heap verification techniques and prepare a large-scale grant proposal for developing novel software technologies and tools to represent, analyse and reason about pointer programs, which shall be submitted to DFG/EPSRC in February 2013. That proposal will address and further elaborate the first and third goals above.

The second goal above shall already be researched during two joint workshops and several mutual visits, which shall be funded under this proposal and by matching funds already granted to us by the University of York. One shortcoming of many existing approaches to heap analysis and verification is that they expose irrelevant detail to the analyst, making it difficult to identify faults. One of our research goals is to develop new mathematical ways of representing heaps and analyzing them, to obscure irrelevant detail and let the analyst focus on things that are really important. We will design a custom, domain-specific language for reasoning about heap structures and operations, which will allow for a new set of tools expressly for representing heaps and operations thereupon. This language will be defined in terms of a meta-model, and its semantics will be given by model transformations or graph-grammar rules; these rules can also be used to specify the heap manipulating programs. This will enable researchers to integrate several heap verification techniques and tools, and will thus make it easier for programmers to track down memory faults.

Collaboration concept. This grant application shall initiate a bilateral Bamberg/York research initiative in the field of heap analysis and verification, on the basis of the following concept.

Two bilateral short workshops shall give researchers from Bamberg and York a platform for exchanging ideas. The first workshop will be organized and fully funded by York and take place in June/July 2012 in York. It will identify state-of-the-art technologies that can contribute to our initiative. The second workshop will be hosted in October/November 2012 at Bamberg, for which funding is sought as part of this grant application. It will focus on developing the structure, objectives and methodology for a large-scale joint grant proposal to the DFG and UK's EPSRC, which shall be submitted in February 2013. To prepare and coordinate this proposal, several short, mutual visits between the senior researchers are required. York has already secured funding for a visit of the two senior researchers of Bamberg to York, while this application seeks funding for the visits of York's senior researchers to Bamberg. In the long term, it is expected that the bilateral collaboration will grow into a sustainable international research network on heap verification (with funding for research visits), which shall be supported by grants of the European Commission.

2.2 Bilateral Workshop

A two-day workshop at Bamberg in October/November 2012 will follow the one at York, which introduced the researchers from both sides, discussed their expertise on the state-of-the-art in heap analysis and verification, and gathered requirements for a domain-specific language for representing and reasoning about heap structures and heap operations.

The Bamberg workshop has two purposes. Firstly, researchers shall follow-up on the results of the York workshop, by exchanging and reporting on ideas for a prototype of the desired domain-specific language. Secondly, they shall update their York talks on the basis of the synergies identified at that workshop, showing how their line of research can help the envisaged joint Bamberg/York research project and concretising ideas how people could best work together. The outcome shall be a draft research plan, including goals and objectives, identified research directions and management aspects, and a prototype heap representation language that will enable researchers to collaborate on tool building and exchange case studies.

In addition, two internationally leading researchers outside Bamberg and York have agreed to participate in the Bamberg workshop, if their schedules permit: Prof. Reinhard Wilhelm of Saarland University, Germany, and Prof. Peter O'Hearn of University College London, U.K. Both are very experienced researchers whose expertise comprises valuable techniques to heap analysis and verification that complement the ones investigated at Bamberg and York. Prof. Wilhelm has worked on shape graphs for heap abstraction and evolution logic for heap verification, while Prof. O'Hearn is a specialist in shape analysis, abduction for heap abstraction, and data refinement in the presence of pointers. Both researchers will help us to validate that the research gaps identified by us are significant and that our research direction promises high impact.

Participants from Germany

Prof. Gerald Lüttgen (gerald.luettgen@swt-bamberg.de)
Prof. Michael Mendler (michael.mendler@uni-bamberg.de)
Dr. Joaquin Aguado (joaquin.aguado@uni-bamberg.de)
Dr. Stefan Rieger (stefan.rieger@swt-bamberg.de)
Dr. David White (david.white@swt-bamberg,de)

These researchers are affiliated with the Faculty of Information Systems and Applied Computer Science, University of Bamberg, 96045 Bamberg, Germany. Prof. Lüttgen is Head of the Software Technologies Group, where Dr. Rieger and Dr. White are post-docs; Prof. Mendler is Head of the Informatics Theory Group, where Dr. Aguado is a post-doc.

Prof. Dr. Reinhard Wilhelm (wilhelm@cs.uni-saarland.de); Compiler Research Group, FR 6.2 – Informatik, Saarland University, Postfach 15 11 50, 66041 Saarbrücken, Germany.

Participants from Abroad

Prof. Richard Paige (richard.paige@cs.york.ac.uk)
Prof. Colin Runciman (colin.runciman@cs.york.ac.uk)
Prof. Jim Woodcock (jim.woodcock@cs.york.ac.uk)
Dr. Detlef Plump (detlef.plump@cs.york.ac.uk)
One Research Assistant of the researchers above, to be named

These researchers are affiliated with the Department of Computer Science, University of York, Deramore Lane, York YO10 5GH, U.K. Prof. Paige is Head of the Enterprise Systems Group; Prof. Runciman heads the Programming Languages and Systems Group, where Dr. Plump is a Senior Lecturer; Prof. Woodcock is a member of the High Integrity Systems Group.

Prof. Peter O'Hearn (p.ohearn@cs.ucl.ac.uk); Computer Science Department, University College London, Gower Street, London WC1E 6BT, U.K.

Overview of Workshop Programme

The workshop programme consists of (i) one day of short talks, with speakers and topics as listed below; (ii) half day on prototyping our domain-specific language within small working groups and a concluding plenary session; (iii) half day of sketching the joint research proposal.

Speaker	*Topic*
Lüttgen:	Pointer Analysis on Object Code
Mendler:	Logics for Heap Verification
O'Hearn:	Abduction for Heap Abstraction
Paige:	Model Transformations for Heap Verification
Plump:	Separation Logic and Hyperedge Replacement Grammars
Rieger:	Heap Abstraction via Graph Grammars
Runciman:	Specifying Pointer Structures by Graph Reduction
White:	Learning Heap Operations Using Pattern Recognition Techniques
Wilhelm:	Heap Analysis Using Shape Graphs
Woodcock:	Unified Theories for Pointer Programs

2.3 Guest Visits

York will host and fund one one-week visit of Prof. Lüttgen and Prof. Mendler to York in late July 2012. This will focus on corroborating the requirements for the domain-specific language for heap representation and analysis which have been elicited during the first workshop, and on analyzing common aspects of existing techniques for heap analysis and verification.

Here, we request funding for one one-week visit of each of York's senior researchers (Prof. Paige, Dr. Plump, Prof. Runciman and Prof. Woodcock) to Bamberg between November 2012 and February 2013. The visits shall consolidate and elaborate on the ideas on novel techniques for heap analysis and verification that originated from the workshops at York and Bamberg. This includes structuring the joint large-scale research project that shall be proposed for funding to DFG/EPSRC in February 2013, by agreeing on work packages, research methodology and distribution of work. In particular, synergies and approaches to tool building shall be discussed, as toolsets will be an important outcome of our envisaged research project. Identifying suitable tools for heap analysis, or components to be used within them, can help build a common infrastructure for our toolsets. Experiments and concrete case studies for evaluation shall also be identified. All aspects above require the researchers of Bamberg and York to coordinate closely, which can best be achieved via mutual visits.

3 Funds Requested

Funds are requested for one two-day bilaterial workshop to take place at Bamberg in October/November 2012, and for four one-week visits to Bamberg of Prof. Paige, Dr. Plump, Prof. Runciman and Prof. Woodcock, respectively, between November 2012 and February 2013. The costs for these activities are expected to total €10,100 as detailed below.

3.1 Bilateral Workshop

The workshop will have 12 participants and is estimated to cost *€6,100 in total*. This includes, for each of the 6 UK participants, €750 for travel (flight+train: €500; accommodation: €250), and for Prof. Wilhelm of Saarland University €400 for travel (train: €150; accommodation: €250). In addition, room hire, refreshments and lunches will approximately cost €100 per participant.

3.2 Guest Visits

Funds are requested for one one-week visit to Bamberg by each of York's senior researchers (Prof. Paige, Dr. Plump, Prof. Runciman, Prof. Woodcock). The costs are expected to be €1,000 per visit (travel: €500; accommodation: €400; subsistence: €100), thus *totalling €4,000.*

4 Erklärungen (Legal Declarations in German)

4.1 Antrag an anderer Stelle

Ein Antrag auf Finanzierung dieser Maßnahme wurde bei keiner anderen Stelle eingereicht. Wenn ich einen solchen Antrag stelle, werde ich die Deutsche Forschungsgemeinschaft unverzüglich benachrichtigen.

4.2 Regeln guter wissenschaftlicher Praxis

Ich verpflichte mich, mit der Einreichung des Antrags auf Bewilligung einer Beihilfe bei der DFG die Regeln guter wissenschaftlicher Praxis einzuhalten.

5 Signature

30th May 2012

Date Prof. Dr. Gerald Lüttgen

6 List of Appendices

A. Information repeated in German

B. Support letter

C. Curriculum vitae of Prof. Dr. Gerald Lüttgen

D. Curriculum vitae of Prof. Richard Paige, Ph.D.

References

[1] A. Bakewell, D. Plump, and C. Runciman. Checking the shape safety of pointer manipulations. In *Relational and Kleene-Algebraic Methods in Computer Science (RelMiCS 2003)*, vol. 3051 of *LNCS*, pp. 48–61. Springer, 2004.

[2] A. Bakewell, D. Plump, and C. Runciman. Specifying pointer structures by graph reduction. In *Applications of Graph Transformations with Industrial Relevance (AGTIVE 2003)*, vol. 3062 of *LNCS*, pp. 30–44. Springer, 2004.

[3] I. Balaban, A. Pnueli, and L. D. Zuck. Shape analysis by predicate abstraction. In *Verification, Model Checking, and Abstract Interpretation (VMCAI 2005)*, vol. 3385 of *LNCS*, pp. 164–180. Springer, 2005.

[4] J. Berdine, B. Cook, and S. Ishtiaq. SLAyer: Memory safety for systems-level code. In *Computer Aided Verification (CAV 2011)*, vol. 6806 of *LNCS*, pp. 178–183. Springer, 2011.

[5] D. Beyer, T. A. Henzinger, and G. Théoduloz. Lazy shape analysis. In *Computer Aided Verification (CAV 2006)*, vol. 4144 of *LNCS*, pp. 532–546. Springer, 2006.

[6] D. Dams and K. S. Namjoshi. Shape analysis through predicate abstraction and model checking. In *Verification, Model Checking, and Abstract Interpretation (VMCAI 2003)*, vol. 2575 of *LNCS*, pp. 310–323. Springer, 2003.

[7] D. Distefano and M. J. Parkinson. jStar: Towards practical verification for Java. In *Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA 2008)*, pp. 213–226. ACM, 2008.

[8] M. Dodds and D. Plump. Extending C for checking shape safety. *ENTCS*, 154(2):95–112, 2006.

[9] W. Harwood, A. Cavalcanti, and J. Woodcock. A theory of pointers for the UTP. In *Theoret. Aspects of Computing (ICTAC 2008)*, vol. 5160 of *LNCS*, pp. 141–155. Springer, 2008.

[10] J. Heinen, T. Noll, and S. Rieger. Juggrnaut: Graph grammar abstraction for unbounded heap structures. *ENTCS*, 266:93–107, 2010.

[11] O. Lee, H. Yang, and K. Yi. Automatic verification of pointer programs using grammar-based shape analysis. In *European Symp. on Programming (ESOP 2005)*, vol. 3444 of *LNCS*, pp. 124–140. Springer, 2005.

[12] M. Mendler and S. Scheele. Towards a simply typed calculus for semantic knowledge bases. In *Logics, Agents, and Mobility (LAM 2010)*, 2010.

[13] M. Mendler and S. Scheele. Cut-free Gentzen Calculus for multimodal CK. *Inform. and Comput.*, 209(12):1465–1490, 2011.

[14] J.-T. Mühlberg and G. Lüttgen. Verifying compiled file system code. In *Formal Methods (SBMF 2009)*, vol. 5902 of *LNCS*, pp. 306–320. Springer, 2009.

[15] J.-T. Mühlberg and G. Lüttgen. Symbolic object code analysis. In *Model Checking Software (SPIN 2010)*, vol. 6349 of *LNCS*, pp. 4–21. Springer, 2010.

[16] E. M. Nystrom, H.-S. Kim, and W. W. Hwu. Bottom-up and top-down context-sensitive summary-based pointer analysis. In *Static Analysis Symp. (SAS 2004)*, vol. 3148 of *LNCS*, pp. 165–180. Springer, 2004.

[17] P. O'Hearn, H. Yang, and J. C. Reynolds. Separation and information hiding. In *Principles of Programming Languages (POPL 2004)*, pp. 268–280. ACM, 2004.

[18] R. F. Paige and J. S. Ostroff. ERC: An object-oriented refinement calculus for Eiffel. *Formal Asp. Comput.*, 16(1):51–79, 2004.

[19] A. Podelski and T. Wies. Boolean heaps. In *Static Analysis Symp. (SAS 2005)*, vol. 3672 of *LNCS*, pp. 268–283. Springer, 2005.

[20] A. Rensink. Canonical graph shapes. In *European Symp. on Programming (ESOP 2004)*, vol. 2986 of *LNCS*, pp. 401–415. Springer, 2004.

[21] J. C. Reynolds. Separation logic: A logic for shared mutable data structures. In *Logic in Computer Science (LICS 2002)*, pp. 55–74. IEEE, 2002.

[22] S. Rieger. *Verification of Pointer Programs*. PhD thesis, RWTH Aachen University, 2009.

[23] S. Rieger and T. Noll. Abstracting complex data structures by hyperedge replacement. In *Graph Transformations (ICGT 2008)*, vol. 5214 of *LNCS*, pp. 69–83. Springer, 2008.

[24] M. Sagiv, T. Reps, and R. Wilhelm. Parametric shape analysis via 3-valued logic. *ACM TOPLAS*, 24(3):217–298, 2002.

[25] H. Yang, O. Lee, J. Berdine, C. Calcagno, B. Cook, D. Distefano, and P. OHearn. Scalable shape analysis for systems code. In *Computer Aided Verification (CAV 2008)*, vol. 5123 of *LNCS*, pp. 385–398. Springer, 2008.

[26] S. H. Yong and S. Horwitz. Pointer-range analysis. In *Static Analysis Symp. (SAS 2004)*, vol. 3148 of *LNCS*, pp. 133–148. Springer, 2004.

[27] J. Zhu and S. Calman. Symbolic pointer analysis revisited. In *Programming Language Design and Implementation (PLDI 2004)*, pp. 145–157. ACM, 2004.