Jahresbericht 2018 des Lehrstuhls für Informatik 2 (Programmiersysteme)

1 Mitarbeiterinnen und Mitarbeiter

Dipl.-Ing. (FH) Helmut Allendorf (IT-Betreuer), Michael Baer, M. Sc., Dipl.-Inf. Thorsten Blaß, Tobias Feigl, M. Sc., Hon.-Prof. Dr.-Ing. Bernd Hindel, Florian Jung, M. Sc., Marius Kamp, M. Sc., Hon.-Prof. Dr.-Ing. Detlef Kips, Patrick Kreutzer, M. Sc., Florian Mayer, M. Sc., Dipl.-Inf. Daniela Novac, Dr.-Ing. Norbert Oster, Akad. ORat, Prof. Dr. Michael Philippsen (Ordinarius), Prof. em. Dr. Hans Jürgen Schneider (Emeritus), Manfred Uebler (IT-Betreuer), Margit Zenk (Sekretariat)

Gäste am Lehrstuhl: Dr.-Ing. Josef Adersberger, Veronika Dashuber, M. Sc., Dr.-Ing. Martin Jung (Lehrbeauftragter), Dipl.-Math. Jakob Krainz, Florian Lautenschlager, M. Sc., Dr.-Ing. Christopher Mutschler, Dr.-Ing. Klaudia Dussa-Zieger (Lehrbeauftragte)

2 Überblick

Seit Prof. Dr. Michael Philippsen 2002 die Leitung des 1972 gegründeten Lehrstuhls übernommen hat, stehen **Programmiersysteme für heterogene Multicore-Rechner** im Zentrum der Forschungsarbeiten des Lehrstuhls.

Bis 2014/2015 standen dabei vor allem systemnahe Forschungsthemen im Vordergrund, die näher an der Hardware, den Laufzeitsystemen, den unterliegenden Betriebssystemen und der Frage der Programmierung dieser Rechensysteme ausgerichtet waren. Es ging dem Lehrstuhl bisher insbesondere darum, das Parallelisierungspotential einer Anwendung verlustarm an die in der Hardware vorhandene Parallelität anzupassen und die von der Hardware bereitgestellten Möglichkeiten effizient nutzbar zu machen. Weil diese systemnahen Fragen noch immer von aktueller Bedeutung sind, werden sie auch in Zukunft in einigen Projekten des Lehrstuhls bearbeitet.

Jetzt konzentriert sich der Lehrstuhl auf Forschungsthemen, die ingenieurwissenschaftliche Antworten für den Software-Ingenieur liefern, der im Rahmen industrieller Software-Entwicklung für Multicore-Rechner, für daraus bestehende verteilte Systeme, für paralleles Cloud-Computing sowie für vernetzte eingebettete Systeme parallele Software entwickelt.

3 Forschung

AnaCoRe – *Analyse von Code-Repositories*: Bei der Weiterentwicklung von Software führen die Entwickler oftmals sich wiederholende, ähnliche Änderungen durch. Dazu gehört beispielsweise die Behebung von Fehlern in funktional ähnlichen Komponenten oder die Parallelisierung von sequentiellen Programmteilen. Wenn jeder Entwickler die nötigen Änderungen selbst erarbeiten muss, führt dies leicht zu fehlerhaften Programmen, weil weitere zu ändernde Stellen übersehen werden. Wünschenswert wäre stattdessen ein automatisiertes Verfahren, das ähnliche Änderungen erkennt und mit dieser Wissensbasis Software-Entwickler bei weiteren Änderungen unterstützt.
SYFEX ist ein neues Verfahren zur symbolischen Code-Ausführung, welches die Ähnlichkeit des Verhaltens zweier Code-Teilstücke bestimmt. In den Jahren 2017 und 2018 wurde SYFEX optimiert. Des

Weiteren wurde mit der Erstellung eines Datensatzes semantisch ähnlicher Methoden aus quelloffenen Software-Archiven begonnen. Um auch in größeren Code-Projekten nach semantisch ähnlichen Code-Fragmenten suchen zu können, wurde ein neues Verfahren entwickelt, das mit Hilfe der Genetischen Programmierung mehrere Basiskomparatoren hierarchisch verknüpft. Die Implementierung dieses Verfahrens wurde in den Jahren 2017 und 2018 weiter verbessert. Zudem spielte die tiefergehende Evaluation des Verfahrens auf Basis von Methodenpaaren aus Software-Archiven sowie von Abgaben für Programmieraufgaben eine wichtige Rolle.

AuDeRace – *Automatische Erkennung von Wettlaufsituationen*: Moderne Software enthält immer mehr Parallelität. Durch diese Nebenläufigkeit treten etliche neue, teils schwer zu lokalisierende Fehler auf, die nicht mehr durch herkömmliche Testverfahren sicher detektiert werden können. Ob ein solcher Fehler auftritt, hängt von dem konkreten Ausführungsplan der Aktivitätsfäden ab. Dieser unterscheidet sich jedoch bei jeder Ausführung und ist auch stark von dem darunterliegenden System abhängig. Somit treten entsprechende Fehler normalerweise nicht bei jedem Testlauf auf, je nach Testsystem sogar niemals.
Im Jahr 2018 lag der Fokus auf einer deterministischen Ausführung von Testfällen. Es wurde ein Konzept erarbeitet, um reproduzierbare Ergebnisse bei der Ausführung zu erzielen: Ein zusätzlicher Ablaufplan spezifiziert das Ablaufverhalten der verschiedenen Aktivitäten. Eine Instrumentierung von vorher markierten Positionen im Code und anderen relevanten Bytecode-Instruktionen soll hier während der Ausführung entsprechend die Kontrolle an eine Verwaltungsaktivität abgeben, die den Ablauf steuert. Damit die Markierungen (Angabe von Positionen im Quellcode) auch nach Änderungen am Quellcode gültig bleiben, sollen diese automatisch auf eine neue Version angepasst werden können. Eine Merge-Technik ähnlich derer zu Versionsverwaltungssystemen soll hier eingesetzt werden.
Das Projekt stellt einen Beitrag des Lehrstuhls Informatik 2 zum IZ ESI (Embedded Systems Initiative, http://www.esi.fau.de/) dar.

AutoCompTest – *Automatisiertes Testen von Übersetzern*: Übersetzer für Programmiersprachen sind äußerst komplexe Anwendungen, an die hohe Korrektheitsanforderungen gestellt werden: Ist ein Übersetzer fehlerhaft, so generiert dieser u.U. fehlerhaften Code oder stürzt bei der Übersetzung mit einer Fehlermeldung ab. Solche Fehler in Übersetzern sind oftmals schwer zu bemerken oder zu umgehen. In der Praxis scheitert das automatisierte Testen von Übersetzern deshalb oftmals daran, dass kein zugeschnittener Programmgenerator verfügbar ist und die Entwicklung eines solchen einen zu hohen Aufwand bedeutet.
Im Jahr 2018 haben wir mit der Entwicklung eines entsprechenden Werkzeugs begonnen. Als Eingabe dient eine Spezifikation der syntaktischen und semantischen Regeln der jeweiligen Programmiersprache in Form einer abstrakten Attributgrammatik. Eine solche erlaubt eine knappe Notation der Regeln auf hohem Abstraktionsniveau. Ein von uns neu entwickelter Algorithmus erzeugt dann Testprogramme, die allen spezifizierten Regeln genügen. Der Algorithmus nutzt dabei diverse technische Ideen aus, um eine angemessene Laufzeit zu erreichen. Dies ermöglicht die Generierung großer Testfallmengen in vertretbarer Zeit, auch auf üblichen Arbeitsplatzrechnern. Eine erste Evaluation hat nicht nur gezeigt, dass unser Verfahren sowohl effektiv als auch effizient ist, sondern auch dass es flexibel einsetzbar ist. So haben wir mit Hilfe unseres Verfahrens nicht nur Fehler in den C-Übersetzern gcc und clang entdeckt (unser Verfahren erreicht dabei eine ähnliche Fehleraufdeckungsgüte wie ein sprachspezifischer Programmgenerator aus der wissenschaftlichen Literatur), sondern auch diverse Bugs in mehreren SMT-Entscheidern. Einige der von uns entdeckten Fehler waren den jeweiligen Entwicklern zuvor noch unbekannt.

DfD – *Design for Diagnosability*: Viele Software-Systeme verhalten sich während der Testphase oder sogar im Regelbetrieb im negativen Sinne auffällig. Die Diagnose und die Therapie solcher Laufzeitanomalien ist oft langwierig und aufwändig bis hin zu unmöglich. „Design for Diagnosability“ beschreibt eine Werkzeugkette mit Modellierungssprachen, Bausteinen und Werkzeugen, mit denen die Diagnosefähigkeit von Software-Systemen gesteigert wird.
Auch im Jahr 2018 haben wir noch weitere Beiträge im Forschungsprojekt geleistet: Wir haben ein Papier auf der PROFES 2018 veröffentlicht, in dem wir Techniken und Erkenntnisse beschreiben, wie man

Laufzeitdaten in einem großen Software-Projekt schon zur Entwicklungszeit für alle Projektbeteiligten anbieten kann und damit die Zusammenarbeit verbessert. Wir haben das Open Source Projekt von Chronix gewartet und dabei weiter stabilisiert (Aktualisieren von Versionen, Fehlerbehebungen etc.).

GIFzuMINTS – *Grundlagen der Informatik als Fundament eines zukunftsorientierten MINT-Studiums*: Für den Studienerfolg haben sich die in der Studieneingangsphase verorteten Lehrveranstaltungen für viele Studierende als problematische Hürde erwiesen, die letztlich häufig zum Studienabbruch führen kann. Aus diesem Grund widmen wir uns dem Ausbau der Unterstützung von angehenden Studierenden beim Übergang Schule-Hochschule sowie während der Studieneingangsphase.
2018 stand unter anderem die Untersuchung der Wirksamkeit aller im Projekt entwickelten Maßnahmen im Fokus. Untersucht wurde der Einfluss des angestiegenen Übungsgruppenangebots und der umfangreichen Unterstützung durch die Tutorinnen und Tutoren. Weiterhin wurden die Auswirkungen der Teilnahme am Repetitorium auf die Leistungen in den Übungen und in der Klausur untersucht.

Holoware – *Holoware*: Der Aufwand für das Verstehen von Software umfasst in Entwicklungsprojekten bis zu 30% und in Wartungsprojekten bis zu 80% der Programmieraufwände. Die dreidimensionale Visualisierung von Software steigert das Verständnis der Sachverhalte deutlich, und damit liegt eine Nutzung von Virtual-Reality-Techniken nahe.
Seit dem Projektstart im September 2018 konnten bereits wesentliche Beiträge geleistet werden. Zunächst wurde ein funktionsfähiger VR-Visualisierungsprototyp zu Demonstrations- und Forschungszwecken entwickelt. Als Grundlage für deren Analyse und Visualisierung wurden anschließend dynamische Laufzeitdaten auf die statische Struktur abgebildet. Schließlich wurde ein Ansatz zur Anomalieerkennung von Ablaufspuren durch ein Unsupervised-Learning-Verfahren entworfen und implementiert.

ICPC – *International Collegiate Programming Contest an der FAU*: Seit 1977 wird der International Collegiate Programming Contest (ICPC) ausgetragen. Dabei sollen Teams aus je drei Studierenden ca. 13 Programmieraufgaben lösen. Als Erschwernis kommt hinzu, dass nur ein Computer pro Gruppe zur Verfügung steht.
Im Jahr 2018 fanden zwei lokale Wettbewerbe an der FAU statt. Im Wintersemester wurde ein Mannschaftswettbewerb ausgetragen mit dem Ziel, neue Studierende für die Wettbewerbe zu begeistern – es meldeten sich 30 Erlanger Teams an. Jedes Team bestand aus maximal drei Studierende. Außerdem nahmen noch 57 Teams von anderen deutschen Universitäten teil. Im Sommersemester fand vor dem Wettbewerb zum wiederholten Mal das Hauptseminar *Hallo Welt! - Programmieren für Fortgeschrittene* statt, um Studierende verschiedener Fachrichtungen in Algorithmen und Wettbewerbsaufgaben zu schulen. Der Wettbewerb im Sommersemester diente danach der Auswahl der studentischen Vertreter der FAU für den NWERC 2018 in Eindhoven (NL). Insgesamt nahmen an dem deutschlandweit organisierten Ausscheidungskampf 30 Teams der FAU mit Studierende verschiedenster Fachrichtungen teil. Aus den besten Teams und unter Berücksichtigung der Altersbegrenzung, wurden neun Studierende ausgewählt, die für den NWERC Dreierteams bildeten. Diese erreichten dann beim NWERC in Eindhoven die Plätze 53, 57 und 63 von insgesamt 119 teilnehmenden Teams.

ORKA-HPC – *OpenMP für rekonfigurierbare heterogene Architekturen*: High-Performance Computing (HPC) ist ein wichtiger Bestandteil für die europäische Innovationskapazität und wird auch als ein Baustein bei der Digitalisierung der europäischen Industrie gesehen. Rekonfigurierbare Technologien wie Field Programmable Gate Array (FPGA) Module gewinnen hier wegen ihrer Energieeffizienz, Performance und ihrer Flexibilität immer größere Bedeutung.
Im Jahr 2018 wurde ein source-to-source Übersetzerprototyp für die Umschreibung von OpenMP-C-Quellcode entwickelt. Darüber hinaus wurde ein HLS-Übersetzerprototyp implementiert, der in der Lage ist, C-Code in Hardware zu übersetzen. Außerdem wurden mehrere experimentelle FPGA-Infrastrukturen

für die Ausführung von Beschleunigerkernen entworfen und umgesetzt.

ParCAn – *Parallele Code-Analyse auf einer GPU*: Im Übersetzerbau (und auch an anderen Stellen) gibt es Analyseverfahren, bei denen Informationen solange durch einen Graph propagiert und dabei verändert werden, bis sich das Analyseergebnis als Fixpunkt einstellt. In diesem Projekt entwickeln wir den Programmrahmen ParCAn, in dem verschiedene derartige Verfahren parallel und dadurch schneller auf der Graphikkarte ablaufen können.
Im Berichtszeitraum 2018 schlossen wir die im Vorjahr begonnen Arbeiten an einer Studie zur Effizienz von Graphstrukturen auf der GPU ab und haben weitere Optimierungen an ParCAn vorgenommen. Um die Leistungsfähigkeit unseres Frameworks zu untersuchen, haben wir es in das LLVM-Übersetzersystem integriert. Umfangreiche Vergleichsmessungen zeigten, dass wir mit ParCAn den LLVM-Übersetzungsprozess um bis zu 40% beschleunigen konnten. Die zugehörige Publikation wurde auf einer Fachkonferenz (28th International Conference on Compiler Construction) als Bestes Papier ausgezeichnet.

RuNN – *Rekurrente Neuronale Netze (RNNs) zur echtzeitnahen Bestimmung nichtlinearer Bewegungsmodelle*: Mit wachsender Verfügbarkeit von Information über eine Umgebung (z.B. eine Sporthalle) und über die Objekte darin (z.B. Sportler in der Halle), steigt das Interesse, diese Informationen gewinnbringend zusammenzuführen (sog. Information Fusion) und zu verarbeiten. Zum Beispiel will man physikalisch korrekte Animationen (z.B. in der virtuellen Realität) von komplexen und hochdynamischen Bewegungen (z.B. in Sportsituationen) in Echtzeit rekonstruieren. Das Kernziel des Projekts ist es, zu evaluieren, wie Methoden des maschinellen Lernens zur Beschreibung von komplexen und nichtlinearen Bewegungen eingesetzt werden können. Dabei soll untersucht werden, ob RNNs die Bewegungen eines Objektes physikalisch korrekt beschreiben und bisherige Methoden unterstützen oder ersetzen können.
Im Jahr 2018 wurde zunächst ein tieferes Verständnis der Ausgangssituation aufgebaut. Dabei wurden Methoden des maschinellen und tiefen Lernens zur Bewegungserfassung, Bewegungsrekonstruktion und Merkmalsextraktion untersucht. Die dabei gewonnenen Erkenntnisse konnten genutzt werden, um sogenannte relative Pedestrian Dead Reckoning (PDR) Verfahren mit Hilfe von Bewegungsklassifizierern zu stabilisieren. Das tiefere Funksignalverständnis ermöglichte das Abbilden von Langzeitfehlern in RNN-basierten Bewegungsmodellen, um die Positionsgenauigkeit und Stabilität zu verbessern und nahe Echtzeit vorherzusagen. Ebenfalls wurde im Rahmen einer Großstudie mit sozialwissenschaftlichem Hintergrund das weltgrößte virtuelle Dinosauriermuseum eröffnet. Es konnte gezeigt werden, dass ein auf das Einsatzszenario optimiertes Bewegungsmodell die menschliche Bewegung robust und genau (i.S.v. kein signifikanter Einfluss auf die Simulatorkrankheit) vorhersagen kann.

SoftWater – *Software-Wasserzeichen*: Unter Software-Wasserzeichnen versteht man das Verstecken von ausgewählten Merkmalen in Programme, um sie entweder zu identifizieren oder zu authentifizieren. Das ist nützlich im Rahmen der Bekämpfung von Software-Piraterie, aber auch um die rechtmäßige Nutzung von Open-Source Projekten zu überprüfen. Ziel unseres Forschungsprojekts ist es, ein Wasserzeichen-Framework zu entwickeln, dessen Verfahren automatisiert (insbesondere ohne Beitrag der Programmierer zum Einbettungsprozess) beim Übersetzen des Programms Wasserzeichen hinzufügen.
Im Jahr 2018 wurden im Rahmen von zwei Bachelorarbeiten Methoden zur symbolischen Ausführung und Funktionssynthese untersucht, um zu ermitteln, welche sich für unseren Ansatz am Besten eignet.

4 Lehre

Der Lehrstuhl für Programmiersysteme bietet im Wintersemester das Pflichtmodul *Algorithmen und Datenstrukturen (AuD)* und im Sommersemester *Parallele und Funktionale Programmierung (PFP)* an. Da diese Module fakultätsübergreifend auch anderen Studiengängen (insbesondere Wirtschaftsinformatik

bzw. International Information Systems, Informations- und Kommunikationstechnik, Mathematik u.v.a.) angeboten werden, erreichten die Hörerzahlen mit 773 (AuD im WS2017/18) bzw. 382 (PFP im SS2018) im Berichtszeitraum jeweils einen Allzeitrekord seit der Einführung der Module, die sich schließlich auch in der hohen Zahl an Prüfungsanmeldungen (712 in AuD bzw. 329 in PFP) niederschlagen. In der Vertiefungsrichtung Programmiersysteme bietet der Lehrstuhl verschiedene Module zu den Themen *Übersetzerbau*, *Clustercomputing* und *Testen von Softwaresystemen* an. Die Seminare *Hallo Welt! für Fortgeschrittene* und *Machine Learning* waren ebenfalls innerhalb kürzester Zeit restlos ausgebucht.
Insgesamt betreute der Lehrstuhl für Programmiersysteme im Berichtsjahr fünf Masterarbeiten, drei Masterprojekte und drei Bachelorarbeiten.

5 Publikationen

[1] Lugrin JL., Kern F., Schmidt R., Kleinbeck C., Roth D., Daxer C., Feigl T., Mutschler C., Latoschik ME.: A Location-Based VR Museum. In *10th International Conference on Virtual Worlds and Games for Serious Applications (VS Games 2018)*. 2018. doi:10.1109/VS-Games.2018.8493404

[2] Roth D., Kleinbeck C., Feigl T., Mutschler C., Latoschik ME.: Beyond Replication: Augmenting Social Behaviors in Multi-User Social Virtual Realities In *25th IEEE Conference on Virtual Reality and 3D User Interfaces (IEEE VR 2018)* 2018. doi:10.1109/VR.2018.8447550

[3] Feigl T., Mutschler C., Philippsen M.: Head-to-Body-Pose Classification in No-Pose VR Tracking Systems. In *25th IEEE Conference on Virtual Reality and 3D User Interfaces (IEEE VR 2018)*. 2018. doi:10.1109/VR.2018.8446495

[4] Feigl T., Mutschler C., Philippsen M.: Human Compensation Strategies for Orientation Drifts. In *25th IEEE Conference on Virtual Reality and 3D User Interfaces*. 2018. doi:10.1109/VR.2018.8446300

[5] Lautenschlager F., Ciolkowski M.: Making Runtime Data Useful for Incident Diagnosis: An Experience Report. In *International Conference on Product-Focused Software Process Improvement (PROFES 2018)*. 2018. doi:10.1007/978-3-030-03673-7_33

[6] Gorse L., Löffler C., Mutschler C., Philippsen M.: Optical Camera Communication for Active Marker Identification in Camera-based Positioning Systems. In *15th Workshop on Positioning, Navigation and Communications (WPNC'18)*. 2018. doi:10.1109/WPNC.2018.8555846

[7] Feigl T., Nowak T., Philippsen M., Edelhäußer T., Mutschler C.: Recurrent Neural Networks on Drifting Time-of-Flight Measurements. In *9th International Conference on Indoor Positioning and Indoor Navigation (IPIN 2018)*. 2018. doi:10.1109/IPIN.2018.8533813

[8] Feigl T., Mutschler C., Philippsen M.: Supervised Learning for Yaw Orientation Estimation. In *Proceedings of the 9th International Conference on Indoor Positioning and Indoor Navigation (IPIN 2018)*. 2018. doi:10.1109/IPIN.2018.8533811