Informatik-Kolloquium

Fachgruppe Informatik

Lehrstuhl Informatik 2

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

EINLADUNG

Zeit: Donnerstag, 16.04.2009, 11.00 Uhr

Ort: Seminarraum Informatik 11, Ahornstr. 55
(Altbau, 3. OG, Raum 2323)

Referent: Herr Dr. Clemens Grelck,
University of Amsterdam, NL
University of Hertfordshire, UK

Titel: From Implicit Concurrency Utilisation to Explicit Concurrency Engineering: SaC and S-Net

Abstract:

The first generation of multicore processors has reached the mass computer market while the development roadmaps of all major processor manufacturers promise a steep rise in the number of cores for the near future. This paradigm shift in processor architecture brings parallel computing from the niche market of supercomputing applications into the mainstream of computing. These new hardware requirements will cause a similar paradigm shift in software engineering: To make efficient use of available hardware resources, any software must become parallel. Established parallel programming models and languages, however, are very much geared towards the needs of supercomputing applications: They allow expert programmers to achieve high runtime efficiency on a given class of machines, but incur development costs that are hardly justifiable outside supercomputing labs.

We present two complementary approaches to efficient programming of multicore systems: SaC and S-Net. SaC is a fully-fledged functional array programming language that adopts the syntactic conventions of C/C++/C#/Java to allow for a smooth transition from these languages. The array programming model of SaC encourages a data parallel programming style that is suitable for fully compiler-based identification and utilisation of concurrency. In essence, programmers may write a parallel program with a sequential mindset. Our aggressively optimising SaC compiler yields sequential programs that are competitive in performance with C and Fortran codes; automatic parallelisation achieves true speedups over these machine-oriented languages.

Not all problems lend themselves to a data parallel solution in a natural way. Automatic detection of other forms of concurrency in regular sequential code, however, has proven to be be overly ambitious. Therefore, we have proposed S-Net as a declarative coordination language for explicit concurrency engineering. S-Net achieves a near complete separation of concerns between the development of sequential (or implicitly parallel) application building blocks and their parallel orchestration in a streaming network of asynchronous components. Concurrency is dealt with explicitly, but instead of augmenting a sequential language with all sorts of annotations and, hence, intertwining computing and concurrency issues, we introduce an explicit concurrency layer into the software engineering process.

Es laden ein: Die Dozenten der Informatik