Future Distributed Applications Based on Mobile Cloud Computing and Software-Defined Networks

Prof. Dr. Sergei Gorlatch
University of Muenster, Germany

We consider an emerging class of challenging networked applications called Real-Time Online Interactive Applications (ROIA). ROIA are networked applications connecting a potentially very high number of users who interact with the application and with each other in real time, i.e., a response to a user's action happens virtually immediately. Typical representatives of ROIA are multiplayer online computer games, advanced simulation-based e-learning and serious gaming. All these applications are characterized by high performance and QoS requirements, such as: short response times to user inputs (about 0.1-1.5 s); frequent state updates (up to 100 Hz); large and frequently changing numbers of users in a single application instance (up to tens of thousands simultaneous users).

This talk will address two challenging aspects of future Internet-based ROIA applications: a) using Mobile Cloud Computing for allowing high application performance when a ROIA application is accessed from multiple mobile devices, and b) managing dynamic QoS requirements of ROIA applications by employing the emerging technology of Software-Defined Networking (SDN).

Towards High-Level Programming for Many-Core Systems Sergei Gorlatch, University of Muenster, Germany Application development for modern high-performance systems with many cores, i.e., comprising multiple Graphics Processing Units (GPUs) and multi-core CPUs, currently exploits low-level programming approaches like CUDA and OpenCL, which leads to complex, lengthy and error-prone programs.

In this talk, we advocate a high-level programming approach for such systems, which relies on the following two main principles: a) the model is based on the current OpenCL standard, such that programs remain portable across various many-core systems, independently of the vendor, and all low-level code optimizations can be applied; b) the model extends OpenCL with three high-level features which simplify many-core programming and are automatically translated by the system into efficient OpenCL code. The additional high-level features are as follows: 1) memory management is simplified and automated using parallel container data types (vectors and matrices); 2) an automatic data (re)distribution mechanism allows for implicit data movements between GPUs and ensures scalability on multiple GPUs; 3) computations are conveniently expressed using parallel algorithmic patterns (skeletors). The well-defined skeletors allow for formal transformations of SkelCL programs which are used both in the process of program development and in the compilation and optimization phase.

We demonstrate how our programming model and its implementation are used to express parallel applications on one- and two-dimensional data, and we report experimental results that confirm greatly reduced programming effort and program size, as well as competitive target performance of our approach.