Published by Oxford University Press on behalf of The British Computer Society. All rights reserved.
For Permissions, please email: journals.permissions@oupjournals.org
doi:10.1093/comjnl/bxh137

Capsule Reviews

FAIROUZ KAMAREDDINE

The Capsule Reviews are intended to provide a short succinct review of each paper in the issue, in order to bring the content to a wider readership. This issue's Capsule Reviews were compiled by Fairouz Kamareddine. Professor Kamareddine is an Associate Editor of *The Computer Journal* and is based in the School of Mathematical and Computer Sciences at Heriot-Watt University, Edinburgh, UK.

Generalized Bottom Up Parsers With Reduced Stack Activity. ELIZABETH SCOTT AND ADRIAN JOHNSTONE
The best currently known general parsing algorithms are order n^3. The standard LR parsing algorithm allows linear parsing of LR(1) grammars but programming languages are not LR(1) and moreover, an LR(1) grammer can become non-LR(1) when some extra actions are added to the grammar. Natural languages use grammars that are not LR(1), and in natural language parsing, general parsing algorithms have been developed that can deal better with the ambiguities of natural language. The GLR algorithm proposed by Tomita is a generalization of the standard LR algorithm based on the observation of a particular parsing algorithm from the field of natural language parsing. GLR is attractive for programming language design and allows addressing the problems of non-LR(1) grammars without losing the ability to treat LR(1) grammars. However, the GLR algorithm is worst case unbounded polynomial. Aycock and Horspool described an approach where larger regular sublanguages are located reducing stack calls in the Tomita style parsing. However, both Tomita's algorithm and Aycock and Horspool's algorithm fail to terminate if the grammar contains hidden left recursion. This paper describes the so-called Reduction Incorporated Generalized LR (RIGLR) algorithm which is motivated by Aycock and Horspool's strategy, but which terminates and is correct for all context-free grammars. The formal proof of correctness is given in the paper as well as a parser version of the algorithm.

Practical Compiler Techniques on Efficient Multi-threaded code Generation for OpenMP Programs. XINMIN TIAN, M. GIRKAR, A. BIK AND HIDEKI SAITO
OpenMP is a parallel programming model which facilitates parallelization of applications in a portable manner. This paper surveys practical compiler optimization techniques, and the implications of their use with programs developed using the OpenMP model. Issues relating to optimization of C++ and Fortran codes are discussed, as are benchmark software standards based on Pentium and Itanium processor systems.

Taxonomy of Distributed Event-Based Programming Systems. MEIER AND CAHILL
Event-based communication is an important paradigm for asynchronously interconnecting the components in a distributed and heterogeneous environment, and allows components to interact anonymously. Event-based middleware and models are currently being applied to a wide range of application domains. This paper presents a survey of existing event systems structured as a taxonomy of distributed event-based programming systems. A taxonomy presents a set of generic event system properties and can be used to classify any distributed event-based programming system. The root of the taxonomy defines the relationship between an event system, an event service and an event model. Furthermore, the event service is classified according to its organization and interaction model and the taxonomy considers functional and non-functional properties including security, mobility and quality of service. The hierarchical structure on which the taxonomy is based may easily cope with the advances of event systems and their extensions with additional novel properties.