Capsule Reviews

In this issue we introduce a new section of the *Journal* called Capsule Reviews. It is intended to provide a short succinct review of each paper in the issue, in order to bring the content to a wider readership. All reviews here are by Associate Editor, Fairouz Kamareddine.

Using a distributed approach to retrieve and integrate information from heterogenous distributed databases. H. T. EL-KHATIB, M. H. WILLIAMS, D. H. MARWICK AND L. M. MACKINNON

This paper, as the title explains, deals with information integration and retrieval from a collection of heterogeneous distributed databases. It is based on the results of the authors of the MIPS system, which links together distributed heterogeneous databases. This paper has re-engineered the MIPS part of breaking down a query to sub-queries and finding answers to these sub-queries in order to produce a distributed agent-based approach that can handle a larger number of databases. In doing so, the paper describes an agent-based architecture as an alternative to the centralized approach of MIPS. In the authors' view, moving the knowledge from the centralized knowledge base system module and distributing it over the network using software agents, significantly raises the limit on the number of databases that can be handled by the system.

The design is overviewed thoroughly in the paper and the agent structure is given in detail. The authors then illustrate the roles of agents in an example of a simple query ('find the medication price for patient number 529'). The paper is an enjoyable read and has a thorough list of references.

An extended temporal logic for CSCW. C. PAPADOPOULOS

This paper is concerned with CSCW (computer supported cooperative work). Formal specification and verification methods have played a vital role in guaranteeing the correctness of CSCW. This is vital because CSCW systems are often informally described because they are large and involve a large amount of information in various forms. This paper follows the trend of using formal methods to represent and reason about CSCW. It chooses a temporal logic to formally describe properties of CSCW where it develops a groupware temporal logic (GTL) for specifying timing properties of CSCW. It shows that GTL is expressive and develops finite-state models for synchronous and asynchronous collaboration.

The paper concentrates on the importance of facilitating the design of real-life systems and hence the GTL logic it provides builds on many other logics that have proved useful for various important applications (e.g. CTL*, linear-time logic, real-time temporal logic TPTL, real-time event logic SREL). The paper gives the model of GTL and explains in detail unary and binary temporal operators. After giving the logic and its model, the paper develops finite state models for synchronous and asynchronous collaboration. Thereafter, it explains how verification of CSCW with model checking can be carried out to verify the correct behaviour of a CSCW system especially for fairness and flexible sharing. Model checking is used as a method for verification due to its efficiency. The paper is enjoyable to read.

Explaining polymorphic types. Y. JUN, G. MICHAELSON AND P. TRINDER

Type theory has been invented to avoid serious flaws in logic. However, type theory has also played a vital role in the design and implementation of programming languages. Types in programming languages avoid undesired/erroneous computations such as adding a number to a Boolean. There is a variety of typing notions. The authors concentrate on the polymorphic notion. Polymorphic types are a powerful mechanism to avoid unnecessary repetitions and to enable expressive programs. Implementing polymorphic types involves operations of unification and substitution with intermediate results that are not intuitive to the average programmer. For this reason, there has been a large amount of interest in the community in providing help to the programmer to explain the cause of errors during type checking a polymorphic term. This paper focuses on the difference between machine and human explanation of types. It concentrates on a small functional programming language (the λ calculus with let expressions). It starts by explaining the Hindley–Milner algorithm W. From W, the paper moves to a new type inference, algorithm H, which annotates types with information about how they were inferred. H is used to construct an annotated type graph based on the abstract syntax tree recording a history of inference of each node. The annotated abstract syntax tree is used to generate an explanation for the errors. The authors check their method and algorithm via a number of tests and users. Results of these tests are reported on in the paper.

Periodic and aperiodic task scheduling in strongly partitioned integrated real-time systems. D. KIM AND Y.-H. LEE

In integrated real-time systems such as the Integrated Modular Avionics, sharing resources is a key issue. This makes it vital to have a well thought out spatial partitioning which guarantees controlled access to all components of the system. Similarly to space partitioning, temporal partitioning is vital as it guarantees control during hazardous or unexpected events. This paper deals with task scheduling in strongly partitioned integrated real-time systems. It provides a strongly partitioned real-time system (SPIRIT) with a two-level hierarchical scheduling

mechanism used to schedule processors. As a result of this partitioning, performance can be enhanced, and complex systems can be upgraded and easily maintained.

This paper is thorough in presenting the SPIRIT model and its fundamental scheduling theory using the two-layer scheduling approach. Scheduling of both soft and hard aperiodic tasks is well situated within the SPIRIT environment. For example, to dynamically schedule aperiodic tasks in a distance-constrained cyclic schedule, three basic operations to manipulate table entries are proposed: Left-sliding (which slides to the left the current phase of the cyclic schedule), Right-putting (which exchanges the allocation of the current partition) and compacting operations (which reduces the problem of segmentation). The system is well evaluated by simulation studies which judge the response time and acceptance rate.

Concurrency control using timestamp ordering in broadcast environments. V. C. S. LEE, K.-W. LAM AND S. H. SON

The physical constraints of wireless communications pose a number of challenging issues in transaction processing. For example, in wireless mobile networks, the servers have a high bandwidth broadcast capability while the mobile client has a limited bandwidth broadcast capability. Moreover, the large number of mobile clients may overload the server. This paper departs from these observations and uses the so-called broadcast-based data dissemination methods where the server repeatedly and continuously broadcasts all data objects in the database. This paper proposes concurrency control protocols to be applied in such a broadcast-based architecture. The main observation of the paper is that the workloads of many applications in mobile computing environments are comprised of read-only transactions which do not modify data (e.g. stock prices, weather information, etc.). Hence, the proposed protocols of this paper exploit the semantics of read-only transactions. In order to guarantee the correctness and consistency of the processing of transactions, this paper uses optimistic concurrency control with a forward validation protocol. In optimistic concurrency control, transactions are allowed to execute unhindered until they reach their commit point at which they are validated. The paper is well written. The proposed model is well motivated and followed by a series of simulation experiments and discussions of its performance. The experiments highlight the usefulness of the approach of this paper where the read-only nature of some information is exploited and where serialization and other concurrency control methods are brought together to provide nice features that meet the challenges posed by dealing with mobile wireless communications.

Lower bounds for one-to-one packet routing on trees using hot-potato algorithms. A. ROBERTS, A. SYMVONIS AND D. R. WOOD

Before explaining what a one-to-one packet routing problem is, let us first explain what a packet routing problem is. We start with a synchronous network that is represented by a connected undirected graph and a set of packets distributed over the nodes of the graph. Each packet has an origin and a destination node. The packet routing problem is to route each packet to its destination in as few steps as possible, assuming that each edge carries at most one packet in each direction at each time step. The routing pattern is the distribution of the origins and destinations of the packets. Such a pattern is many-to-many if each node could be the origin and destination of more than one packet. The pattern is one-to-one if each node is the origin and destination of at most one packet. Routing algorithms can be on-line or off-line. This paper is concerned with the on-line one-to-one packet routing on trees. It considers a particular algorithm known as the 'hot-potato algorithm' where each packet must traverse a link at every step until it reaches its destination. The paper considers the greedy form of the hot-potato algorithm, which attempts to advance each packet towards its destination. It attempts to bridge the gap concerning the performance analysis of the greedy hot-potato algorithm. The main results of the paper are that: (i) any greedy hot-potato algorithm routes a one-to-one routing pattern on an n-node tree within $2(n-1)$ steps; (ii) that there are one-to-one routing problems requiring at least $3n/2$ steps by an oblivious greedy hot-potato algorithm; and (iii) that the upper bound is optimal. This a very enjoyable, well-written paper where the notions are well presented and explained, and the results are very useful. Very recommended reading.