PARALLEL COMPUTING FOR COMMERCIAL APPLICATIONS

N HOLT

Rapporteur: M Pakzad

PARALLEL COMPUTING FOR COMMERCIAL APPLICATIONS

Nic Holt

ICL, Wenlock Way, West Gorton, Manchester

M12 5DR

Introduction

In the commercial world the requirements of IT systems are dominated by the (usually) critical role that IT systems play in the operational support of the organisation and its activities. In particular, such systems are used for 'mission-critical' applications where such attributes as high availablility, reliability & integrity are assumed. For many commercial organisations, IT systems also provide the foundation upon which the processes which support business operations are built; much of the information associated with such systems is, inevitably, shared on a large scale amongst many disparate components of the organisation; this leads to requirements for high concurrency of access coupled with security and integrity of the information.

Precisely because these IT systems are so crucial to the effective functioning of commercial organisations, there is a need to preserve the investment in existing systems, interfaces, information and software; thus the commercial market places great value on the ability to evolve gradually rather than to change everything in a quick revolution. Evolving international standards have a key role to play in assisting this process, by ensuring stability of standard interfaces whilst allowing for innovative implementation of the underlying support for those interfaces.

Parallel computing technology has several attributes which, increasingly, make it very appropriate for the support of commercial applications:

a) Parallel machines offer opportunities for much more cost-effective support of large-scale IT applications; The cost-performance characteristics of the technology used in small and mid-range systems are at least an order of magnitude better than that employed in conventional 'mainframes' or 'supercomputers'. Parallel computing techniques provide the means to exploit the cheaper technology in commercial applications.

b) The inherent scalability of many parallel architectures offers the opportunity to scale IT systems well beyond the size possible with conventional technology; The absolute performance of a single processor is severely constrained by the implementation technology (even though recent application to micro-processors of techniques to realise *internal* parallelism - the super-scalar designs - have breathed new life into sequential execution). Parallel computing enables the support of IT systems whose power requirements may outstrip the normal evolution of hardware technology alone.

c) Homogeneous parallelism provides a level of redundancy which can be used to increase system availability and fault tolerance. As IT systems become increasingly indispensible within the commercial organisations which they support, qualities such as integrity, reliability and availability become crucial issues. Parallel systems provide an opportunity to exploit the redundancy inherent in large scale replication to achieve high levels of availability and error

recovery. Moreover, such systems may be configured in functionally redundant ways (eg Triple-Modular-Redundancy; Disc Arrays) to ensure high integrity (ie good error detection); in such systems, the performance benefits of parallel execution may be merely a by-product! It should equally be pointed out that large-scale parallel systems may introduce new problems and types of failure.

However, for most commercial organisations, the key challenge is finding ways of realising these potential benefits without requiring massive, disruptive changes to existing IT systems; we will therefore place particular emphasis on such exploitations of parallel technology.

The development of parallel processing technology has, until recently, been led largely by the needs of numerically intensive computation (including parallelisation of scientific Fortran) and, from a language perspective, by research into declarative (eg functional, logic) languages whose semantics are independent of detailed execution order. Thus the range of tools to assist in the creation of commercial applications to exploit parallel processing is rather restricted. More recently, the development of '4th generation' application development tools (although usually rather specifically associated with database applications) has improved matters. The wider availability and applicability of such tools and also of designers and programmers trained and skilled in their use will be an important factor in determining how widely and rapidly parallel processing is used in the commercial world.

The remainder of this paper considers examples in the areas of *Information Storage*, *Information Transmission* and *Information Processing*.

Information Storage & Access

One of the areas relevant to commercial computing which can benefit most immediately and invisibly from parallel computing is that of databases. This results from the large-scale concurrency requirements as well as the opportunity provided by non-procedural database access languages such as SQL which, by raising the level at which interactions with the underlying DBMS take place, provide a natural interface to underpin with parallel technology.

Relational DBMSs, whilst offering the user considerable benefits in terms of database funtionality, ease of administration and enhancement potential, are acknowledged to require significantly more processing power than earlier database models. The resulting increased costs are an obstacle to the wider use of relational DBMSs for major applications in the commercial world. Parallel processing techniques can, as discussed above, be exploited to improve the economics of relational DBMSs - more readily, in fact, than for traditional, navigational databases - thus redressing the balance.

There are various ways in which parallelism can be exploited within the DBMS and these have corresponding implications for the underlying hardware architecture. Broadly, parallelism can be exploited in two ways:

a) Inter-transaction parallelism in which parallelism results from the concurrent execution of several transactions: this is typical for high volumes of simple transactions;

b) Intra-transaction parallelism in which a transaction is decomposed into simpler sub-tasks which may be executed concurrently: this is the only way of usefully applying parallelism to low volumes of complex queries (transactions).

At present, most commercially available general purpose parallel databases only support inter-transaction parallelism but within a year or two most will offer intra-transaction parallelism to support 'management queries'. Parallel database systems generally require extremely careful matching to the underlying operating system and hardware and there is a tendency for DBMSs which support parallel execution to be optimised for a class of hardware architectures; this may also imply a bias towards a particular type of workload.

With conventional multi-processors (the low end of the scale of parallelism) the 'shared-store' approach with large main-store database caches has become popular; at the opposite extreme the trend is towards the 'distributed store' approach (with message passing between processors); in either case, discs may be equally accessible to all processors or each locally connected to a particular processor. In this context it is interesting to observe that, to the application designer, the logical distinction between a truly distributed DBMS and a highly parallel database is becoming almost indiscernable.

Information Transmission

Electronic messaging has come of age and it is now quite routine for interactions between organisations (as well as within) to be effected electronically. The routine use of EDI exemplifies this but other examples include the more informal mail (eg X400) services as well as directory services (eg X500) and the growing interest from the PTTs in offering value-added services as an adjunct to the PSTN.

Protocol engines can be very mill-intensive at high bandwidths as well as requiring reasonable responsiveness; this limits the number of such connections which can be handled by a single processor. Fortunately, since communications protocols are, almost inevitably, designed for distributed implementation, there is considerable potential for parallelism. From the system administration viewpoint there are also significant advantages in aggregating message transmission and switching functions into larger 'hubs'; the systems that result can benefit even further from the high performance and availability offered by parallel processing.

There are many cases in which message switching is closely coupled with information storage: store-and-forward messaging & EDI; intelligent networks (in which the network supplier provides additional, value-added services). Such applications place further demands for processing resources as well as access to any shared database.

Information Processing

At the application level there is a broad spectrum of ways in which parallelism may be exploited; one major distinction is whether the parallelism is explicit or implicit.

At one extreme, various application development routes allow the construction of applications using communicating 'threads' or 'remote procedure calls' (the 'remote'ness may be virtual) by the use of parallel programming languages and libraries; such applications may be executed either within a (symmetrical) close-coupled multiprocessing environment or within a specially engineered distributed system; in either case the application designer has to be aware of the parallel nature of the underlying hardware architecture during both high-level design and low-level implementation.

At the other extreme, high-level, non-procedural languages allow applications to be constructed in a manner which abstracts away from issues of concurrency and parallelism. Important examples of such languages are SQL and Prolog; equally, 4th generation application development tools and AI or 'expert system' shells provide the means of creating applications in a style independent of the underlying architecture. Implicit (but coarse-grained) parallelism may also result from the casual high-concurrency execution of largely independent applications on conventional multi-processor systems (with processes executed on different processors) in order to utilise resources more effectively.

Major application areas include:

High concurrency operational applications (usually supported by a DBMS)

Modelling & simulation (eg financial, economic, 'scientific', CAD)

Process control, scheduling (often heterogeneous)

Optimisation / constraint handling (may give super-linear speed-ups!)

CAD/CAM (VLSI CAD, structural analysis, graphics manipulation)

Image processing

Searching - ad-hoc, fuzzy, complex;

Inference / deduction (incl. Logic, Expert Systems & Neural Nets)

Pattern recognition (image, voice), natural language processing

Conclusions

Currently the number of explicitly parallel systems sold is relatively small but in a significant number of areas they are generally accepted as being a natural and strategic evolution path for current systems. It is highly likely that parallel DBMSs will be the spearhead in the commercial environment since database is at the heart of so many commercial operations and parallel implementations of standard DBMSs are becoming available offering full compatibility with sequential versions.

Thus, the benefits of parallel technology are clear and already realisable in a commercial environment. The key to being able to exploit these advantages in a wide range of commercial applications is to do so in a way which evolves and enhances the capabilities of existing IT systems rather than requiring massive (and therefore disruptive) changes. The development of mechanisms for transparent distribution within Open Systems will provide a valuable intermediate step towards systems offering truly large-scale parallelism.

DISCUSSION

Rapporteur: M Pakzad

During the discussion session Mr Holt was asked about applications which do not lend themselves to parallel processing. He said that these are applications which are serial by nature and those which would require an unacceptable amount of synchronisation to ensure correctness. In the latter case the overhead associated with synchronisation is so large that sequential processing is to be preferred. These applications are limited by the speed of such communications.

Mr Barron asked why ICL have not attempted manufacturing parallel computers up to now. Mr Holt answered that it is only recently that standard operating systems (such as UNIX) have been available for parallel systems and the same is the case with parallel languages.

Professor Tanenbaum commented that parallel software exists for some high value applications. Professor Shepherd suggested that powerful single processor machines can be used instead of a parallel processor. Professor Levy mentioned that we are limited by the speed of a processor in a machine and the only way to process faster is to connect two or more of these processors together in some way. He also said that we are always looking for new ways of using processors.