A New Paradigm For Real-Time Database Management (Non-Refereed)

Will Meilander
Kent State Univ., retired
willcm@charter.net

Abstract

Garey and Johnson in their highly respected text write: “There is wide agreement that a problem is not ‘well solved’ until a polynomial time algorithm is known for it.” Such a “well solved” polynomial time algorithm is the subject of this paper. We show the ATC environment and consequently the database and solution to be polynomial.

Real-time database management (RTDB) has for many years been declared NP-hard or NP-complete for a multiprocessor (MP). We agree. Results achieved in practice support that claim. The approach offered here uses the associative processor (AP), in which a static program providing a “well solved” solution for many real-time databases can be realized. A major military program, the US Navy’s E2C AEW system started using this approach in 1983. The AP called “ASPRO” had 2,000 processing elements and showed a measured throughput improvement of 274 times over the dual processor approach it replaced. Equivalent E2C radar tracking performance has not been achieved in our air traffic control system.

The AP is SIMD architecture, as described in my companion abstract “3-D Real-time Database”. However, because any RTDB system requires high speed I/O, the AP has a multidimensional access memory (MDA) that implements the required I/O performance. Many other SIMD machines developed over the past 35 years could not provide adequate I/O for the RTDB requirements.

Why does the AP handle RTDB problems so much better than the MPs currently in use? There are two parts to the answer. First the AP can implement the RTDB as a 3-D structure, suggested in Codd’s twelve rules for relational databases, and the AP does achieve the ACID requirements demanded of any database system. Second the AP eliminates most of the program requirements of current systems.

Among these MP problems are: Dynamic task scheduling, individual processor state assessment, processor task assignment, data broadcast and reduction, memory contention, bus arbitration software, multi-tasking and multi-thread software, processing for mutual exclusion, maintaining sequential consistency, data sorting and indexing, resorting/reindexing as data changes, data pointer management, shared resources, priority inversion, table/record/item data locking, lock management, coherency management (memory, cache) and preemption management. Each of these problems is the result of multiple instructions. None of these problems can exist in the AP single thread instruction program.

The AP can realize very high-speed performance in the execution of DB transactions because it implements a content addressable memory. The AP, a set processor, executes an operation over a set of data with one instruction. Because of this capability, most transactions are implemented in a single instruction stream that is executed once. The MP, on the other hand, must execute many instructions that move data from “common memory” to processors then execute operations on the individual processors and return results to “common memory”. For some operations, like update a set of radar tracks, the AP performance can be in the order of 1,000 times faster than the MP.

Many call the problem NP-hard. This usually implies some exponential increase in the problem. However, when we look at the ATC problem correctly, we see it is entirely polynomial. The problem is bounded. Flights, sensor reports, terminals, controllers, aircraft types, etc are all increasing linearly with a maximum count for each. As a result the time complexity function (tcf) is O(1) for AP functions like the track update mentioned above. Flight plan to track conformance, coordinate transform, track update and flight plan update are O(1) in the AP. In the MP they are O(n). Similarly, report to track correlation, conflict detection, conflict resolution, terrain avoidance, VFR automatic voice advisory and cockpit situation display are O(N) in the AP. In the MP the same operations are O(N^2). Thus with the AP, we see a very large improvement in throughput for a real-time database problem. Let’s reflect on MPs in ATC.

Since 1963 and the first MP specification for ATC, it has been found impossible to develop a desirable static schedule. As a result, many requirements have been unmet. Starting in 1983, a one billion dollar study by two contractors reached the conclusion that AAS requirements couldn’t be met. Nevertheless, FAA went ahead with AAS until its cancellation in 1994. AAS could have been completed and ATC requirements satisfied using an AP like the Navy’s ASPRO. At the same time, the APs simplicity would have been saved several billion dollars.

The single instruction stream, static schedule used in the AP paradigm eliminates many problems endemic in the MP paradigm. We have shown the following for the real-time database environment. Conceptual complexity; the simplest approach to writing and verifying the correctness of RTDB algorithms. Algorithmic complexity; much simpler and more easily understood algorithms for RTDB problems. Time complexity; the fastest running time for a set of algorithms that will predictably solve problems. Space complexity; the smallest space requirement for RTDB problems.