HPEC 2002 abstract

What is Keeping Hard Real-Time Scheduling from Being a Mainstream Technology in the Embedded Multiprocessing Domain Space?

Mr. Daniel M. Lorts
Doctoral Candidate
University of Texas at Dallas
Department of Computer Science
9017 Lakepointe Avenue, Rowlett, TX 75088
Phone (214) 232-7918, dlorts@computer.org

A vast majority of the historical and recent approaches to Hard Real-Time Scheduling, with or without including fault-tolerance, has been limited with the certain non-realistic assumptions in place at the startup of the research. The topic of real-time scheduling is extremely difficult even before the inclusion of hard-time requirements. Adding fault tolerance increased the degree of complexity even further. Most academic research in this area has been limited in scope to fit the research into the allotted time (semester, degree plan, etc.), typically by imposing some boundary conditions. This is not to say that the work is irrelevant or lacks value. Many of the results throughout the years have been expanded into further developments and do provide a technological path of successes and failures to "tune" the research of others.

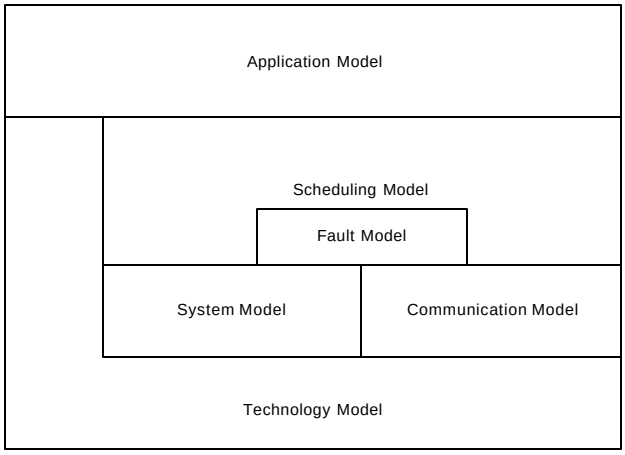

Figure 1. Assumptions Model

The assumptions have been partitioned into six categories as pictured in Figure 1. In the figure, relative associations among the categories are also depicted with outer layers affecting the adjacent inner layers. The six areas identified include System Model, Communication Model, Application Model, Fault Model, Scheduling Model, and Technology Model.

System Model: The system model specifies the architectural aspects of the system including the processing hardware and the interprocessor communication fabric. Because this entity can become quite complex quickly, it often is modeled at a very high level primarily in regards to the processing capabilities of the platform. Often fictitious architectures or topologies are identified to lower the problem complexity. Topologies are assumed to be fully connected.

Communication Model: In this case what is typically done is to assume that the communication costs are negligible or set to some "random" (usually uniform) values.

Fault Model: The fault model, if included at all during a scheduling research effort, is usually too simplistic. It is assumed that there can be only a single fault *somewhere* in the system. This single fault is usually in the execution of a task (or a processor failure) and not within the communication network. When a fault is encountered there is a limited set of recovery techniques including "selective fault tolerance," replication, backups, and redundancy. Temporally, the faults are assumed isolated to a single instance of a task. The measurements of Fault Tolerance levels seem to be inconsistent between various works (QoS issues).

Technology Model: Technology covers a wide variety of entities including the specific hardware assets being utilized to the capabilities of the development tools. In this report, concentration will be placed on the latter. Specifically, in an effort to "optimize" executable modules compilers and linkers are trained to remove unused sections of code. This enhancement could remove any redundant modules inserted to support the fault tolerance policies of the system.

Application Model: When an application is specified for a particular evaluation of a scheduling paradigm the associated application model is either modeled as a directed acyclic graph (DAG) or as a data flow graph (DFG). In either representation, the correctness of the tasks being modeled is compromised. In particular, temporal metrics associated with a task are applied across the entire graph. Other parameters that are often fixed to reduce the complexity include: task preemptability, the number of entry and exit points, data/control dependencies, undefined methods of prioritization, and limited cyclical modeling.

Scheduling Model: Because of the complexity of dynamic scheduling policies most work has been performed developing static scheduling algorithms. This limits the system's adaptability in the event of a fault. Rate monotonic algorithms (RMA) have provided the greatest advances thus far. Metrics associated with the performance of a scheduling algorithm are too generalized and broad scoped.

So where does industry go from here? This research abstract is an attempt to look at some of the areas that have limited the applicability of existing scheduling paradigms. In current research, these areas are being delved into in an attempt to increase the applicability to actual systems and applications. The management of these research efforts needs to transition to a higher level (industry, government research facilities, etc.) instead of the academic community. This would see that there is a continuum of research beyond the aforementioned semester, term, and degree program.

Presentation: Oral
Topic areas:

- Algorithm Mapping to High Performance Architectures

- Reconfigurable Computing for Embedded Systems
- Fault-Tolerant Hardware / Software Techniques