APPLYING STATISTICAL DESIGN TO CONTROL THE RISK OF OVER-DESIGN WITH STOCHASTIC SIMULATION

Yi Wu[1], Peng Zhou[1*], Jian Lin[1,2] ,Wanhua Qiu[1]

[1]Dept. of Economics and Management, Beijing University of Aeronautics & Astronautics, Beijing

*[*1]Email:*shengniao@gmail.com

[2]Institute of Education, Tsinghua University, Beijing

ABSTRACT

By comparing a hard real-time system and a soft real-time system, this article elicits the risk of over-design in soft real-time system designing. To deal with this risk, a novel concept of statistical design is proposed. The statistical design is the process accurately accounting for and mitigating the effects of variation in part geometry and other environmental conditions, while at the same time optimizing a target performance factor. However, statistical design can be a very difficult and complex task when using classical mathematical methods. Thus, a simulation methodology to optimize the design is proposed in order to bridge the gap between real-time analysis and optimization for robust and reliable system design.

Keywords: Percolated stochastic, Simulation optimization, Statistical design

1 INTRODUCTION

Soft Real-time embedded systems, such as battery-operated PDAs etc., have become increasingly popular in our life. Unlike hard real-time counterparts, soft real-time applications are only expected to guarantee "expected delay" over input data space. However, timing is traditionally treated as a hard constraint throughout the system design process. As a result, applications are often pessimistically analyzed for worst case scenarios, and the slowest responses determine their performance. Although this is a necessity for hard real-time applications, soft real-time counterparts can occasionally take longer than the deadline to finish some tasks. They are often expected to guarantee an expected delay (or latency) rather than a worst case execution time over the input data space. For example, embedded multimedia systems require the processing of signal, image, and video data streams in a timely fashion to the end user's satisfaction. Such applications are often characterized by repetitive processing on periodically arriving inputs, such as voice samples or video frames, and the tolerance to occasional deadline (determined by the throughput requirement of the input data streams) misses without being noticed by human visual and auditory systems. In packet audio applications, loss rates of one to ten percent can be tolerated.

As soft real-time systems can tolerate some violations of timing constraints, those methods which guarantee no deadline missing by considering worst-case execution time (WCET) of each task will often lead to over-designed systems that deliver higher performance than necessary at the cost of expensive hardware, higher energy consumption, and other system resources.

There are plenty of studies on the estimation of soft real-time system's probabilistic performance when the application's computation time can be varied. However, most of their goals are to improve the system's performance or to provide probabilistic performance guarantees. Our recent work discusses a percolated stochastic simulation approach to optimize a system's design by taking advantage of a soft real-time application's tolerance to deadline misses. To the best of our knowledge, there is no reported effort on systematically incorporating an application's performance requirements, uncertainties in execution time,

and tolerance for reasonable execution failures to guide rapid and economic optimization of real-time embedded systems.

In this paper, we study the problem of how to integrate such tolerance to deadline misses into the optimization of soft real-time systems. We propose the novel concept of statistical design for multimedia systems and a simulation methodology to optimize the design. Given the execution time distribution of each task, we have developed a simulation algorithm to estimate the probabilistic timing performance and to manage system resources in such a way that the system achieves the required completion ratio probabilistically with a reduced amount of system resources. This method relaxes the rigid hardware requirements for software implementation and eventually avoids over-designing the soft real-time system.

The rest of the paper is organized as follows: Section 2 gives the review of design methods and rules developed by former writers. The overview of our statistical design methodology is discussed in Section 3. Probabilistic timing performance estimation is discussed in Section 4. The Random Critical Path simulation approach is discussed in Section 5. Section 6 introduces our percolated Stochastic simulation approaches. Finally, the conclusion and future study are discussed in section 7.

2 REVIEW OF DESIGN METHODS AND RULES

From the late 1980s, when schedule ability analysis became one of the traditional fields of investigation in real-time systems research, worst-case execution time analysis (WCET analysis) caught the attention of the research community, and the basic methods of how to calculate a safe and tight WCET using static analysis for imperative programming languages (like C) and simple hardware (like MC68000) were presented. In the 1990s, more and more research groups focused on WCET analysis. As a result, substantial progress has been made in this area in a relatively short time.

Together with schedule ability analysis, worst-case execution time analysis (WCET analysis) forms the basis for establishing confidence in the timely operation of a real-time system. WCET analysis does so by computing (upper) bounds for the execution times of the tasks in the system. These bounds are needed for allocating the correct CPU time to the tasks of an application. They form the inputs for schedulability tools, which test whether a given task set is schedulable (and will thus meet the timing requirements of the application) on a given target system.

Figure 1 represents a typical WCET circle of designing procedure. We start with a pool of target system architectures to select from. We first estimate the worst case execution time (WCET). If this estimation meets without ***any*** deadline misses, it is kept, and we go on to the next step, system synthesis and evaluation. If this is not the case, this estimation is transferred to the system architecture pool to find a way of optimization. The next estimation is then given, and a new circle begins.

This method is pessimistic and suitable for developing systems in a “hard real-time” environment, where any deadline miss will be catastrophic. However, it is not suitable for a soft real-time system design. As is discussed in Section 1, such a soft real-time system could bear some response delay or performance loss while a hard real-time system could stand no such delay or loss. Thus, this type of WCET method will lead to an over-design risk, which may bring huge needless time cost and operation cost losses in production.

Several studies on the probabilistic timing performance estimation for soft real-time systems design have been emerging since the late 1990s. The general assumption is that each task’s execution time can be described by a probability density function that can be obtained by applying path analysis and system utilization analysis techniques. In Kalavade and Moghe (1998) the authors extend the scheduling

algorithms and schedule ability analysis methods developed for periodic tasks in order to provide a probabilistic performance guarantee for semi periodic tasks when the total maximum utilization of the tasks on each processor is larger than one. They describe the transform-task method that transforms each semi-periodic task into a periodic task followed by a sporadic task. The method can provide an absolute guarantee for requests with shorter computation times and a probabilistic guarantee for longer requests. In Hua et al. (2003), a performance estimation tool that outputs the exact distribution of the processing delay of each application is introduced. It can help the designers develop multimedia networked systems requiring soft real-time guarantees in a cost efficient manner. Given that the execution time of each task is a discrete random variable, Hu et al. propose a state-based probability metric to evaluate the overall probabilistic timing performance of the entire task set. Their experimental results show that the proposed metric reflects well the timing behavior of systems with independent and/or dependent tasks.

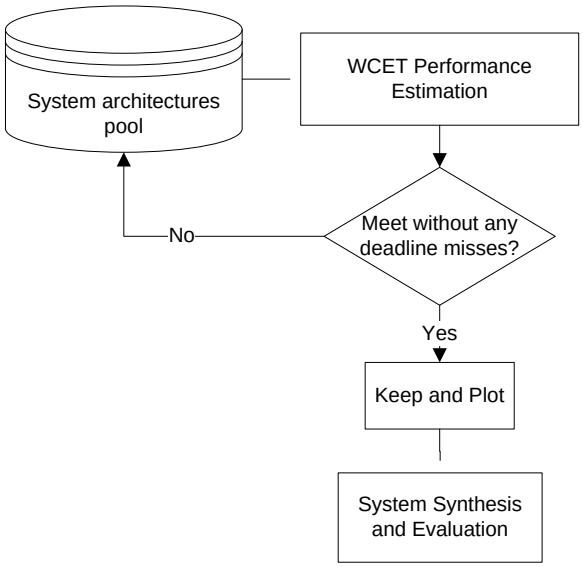

Figure 1. Hard Real-time Design Circle Based On WCET Analysis

3 OVERVIEW OF STATISTICAL DESIGN METHODOLOGY

Considering a soft real-time system, which can tolerate occasional missed deadlines, it is evidently not advisable to apply mechanically those methods and rules used in hard real-time system design. In order to avoid over-designing systems, we propose the concept of "statistical design," where we design the system to meet the timing constraints of periodic applications statistically. That is, the system may not guarantee the completion of every execution or iteration, but it will produce sufficiently many successful completions over a large amount of iterations to meet the user-specific completion ratio. Or even better, the probability that any execution will be completed is not lower than the desired completion ratio.

Clearly, the proposed "statistical design" will be preferred for many embedded soft real-time systems such as portable multimedia systems where high portability, low power consumption and reasonably good performance are equally important.

Figure 2 depicts our statistical design approach for rapid and economic system prototyping. We start with the popular dataflow graph representation of the embedded software, the system's performance requirements (in terms of timing and completion ratio constraints), and a pool of target system architectures to select from. We partition the application into a set of tasks and use profiling tools to collect detailed

execution information for each task. Next, we estimate the system timing performance by task path analysis to check whether it is feasible for the current system configuration to achieve the desired performance. If not, we change the hardware configuration and/or apply software optimization techniques and update the software profiling results that will be used in the next round of system timing performance estimation. We mention that any change to the target hardware configuration and/or software optimization may affect the application's actual execution information, and therefore, the software profiling process will need to be re-started. This iterative design loop terminates when all the design requirements are met. Once the completion ratio constraint is met, we move on to the phase of percolated stochastic simulation optimization. If the constraints can be guaranteed ***with user-specific possibility requires***, the design space of the current profile is kept and plotted. This is the key step in the proposed statistical design optimization where we 1) allocate minimum system resources to each task to make the desired completion ratio probabilistically achievable and 2) develop real time schedulers to manage the resources at run time such that the required completion ratio can be achieved probabilistically. Finally, we conduct system synthesis, simulation, and evaluation before prototyping the system.

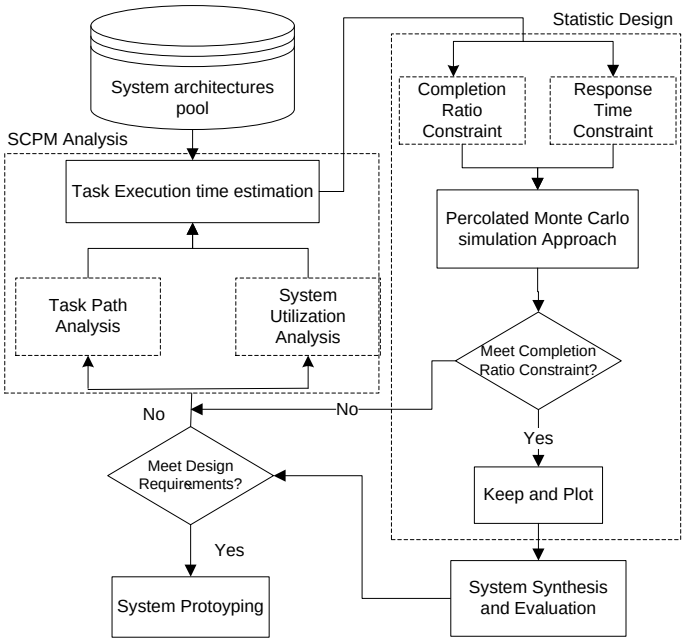

Figure 2. Statistical Design Methodology

In this paper, we will restrict the discussion to one specific part of the statistical design approach – optimization; i.e. we are assuming the existence of a high quality, fast-running simulation model (or meta-model of a simulation model).

4 ESTIMATING THE PROBABILISTIC TIMING PERFORMANCE

In order to determine whether a given system implementation can meet the desired completion ratio constraint, we need to estimate the system's probabilistic timing performance. Specifically, we calculate the upper bound of the completion ratio that the system can achieve to help us in exploring the statistical design space.