Probabilistic Admission Control to Govern Real-Time Systems under Overload

Claude-J. Hamann Michael Roitzsch Lars Reuther Jean Wolter Hermann Härtig

Technische Universität Dresden
Department of Computer Science
Operating Systems Group
drops@os.inf.tu-dresden.de

Abstract

Existing real-time research focuses on how to formulate, model and enforce timeliness guarantees for task sets whose correctness has a temporal aspect. However, the resulting systems often exhibit poor resource utilization due to the resource scheduler reserving more resources than required in order to ensure that admitted schedules can be satisfied under worst case conditions. Weakening the guarantees leads to the known concepts of firm and soft real-time tasks, but we think the paradigm needs to be shifted further, reifying efficient utilization. With Quality-Assuring Scheduling (QAS) we presented such an algorithm. However, its practical applicability is restricted to uniform and harmonic periods, due to its complexity for arbitrary periods. To overcome this limitation, we introduce Quality-Rate-Monotonic Scheduling (QRMS), which, although slightly more pessimistic, is less complex compared to QAS. The admission control is again based on a probabilistic model to ensure that a requested fraction of jobs is successfully executed. Thus, the amount of missed deadlines can be externally controlled, even in sustained overload situations.

1. Introduction

Traditionally, real-time systems are classified into hard and soft real-time systems. While in hard real-time systems, all tasks must always meet their deadlines, soft real-time systems can tolerate some jobs (i.e., an instance of a periodic task) delivering their results after their deadline. The concept of firm tasks provides a mixture of both: Tasks are allowed to miss some deadlines, but unlike soft real-time, the result is worthless when delivered after the deadline, so running jobs beyond their deadline should be avoided. Existing research in this area such as (m,k)-firm tasks [12] provides ways to model the amount of tolerable deadline misses and presents a scheduling algorithm to enforce the requirement. These methods use the relaxed guarantees to increase utilization efficiency, but this is mostly an afterthought. We will follow on this path, but open up a new solution space by using probabilistic guarantees.

The primary obstacle towards efficient resource utilization is that the task model of most real-time admission controls only allows fixed worst-case execution times (WCETs). These often exceed average-case execution times by an order of magnitude. By using the worst-case for admission, a task will get more resources assigned than it needs in average. This observation leads us to the conclusion that the WCET alone is too coarse a description of a task's resource requirements. As such, we explore modeling the execution time of a job using a distribution function. This covers the temporal behavior of the jobs in much greater detail and is obtained as easy as WCET through empiric measurements. With the additional knowledge at hand, the admission control can dedicate less resources to a task while still meeting statistical guarantees, thus reducing resource reservation. Our admission and scheduling policy makes no assumptions regarding properties of this density function. Another algorithm using density functions is Statistical Rate Monotonic Scheduling (SRMS) [2]. However, it requires a trace of the execution times of all jobs *beforehand* for local admission. Our algorithm does not require any temporal knowledge besides the probability distribution.

The approach to describing temporal behavior using a distribution function can be combined with the principle of firm real-time tasks to model systems with statistical guarantees. A given percentage of deadlines as denoted by a quality parameter must be met, missing the remaining is tolerable. Therefore, the fraction of deadlines that are missed cannot only be predicted, but also externally controlled in a straightforward fashion.

Typical execution time distributions have a fairly wide spread. While being disadvantageous for scheduling methods working with worst-case times, our algorithm exploits this behavior to increase utilization efficiency: Even quality levels slightly below 100% can lead to dramatically reduced resource reservation, thus leaving more resources for other tasks. Building on the concept of firm real-time, our method

trades the strictness of the guarantees it provides for a decrease in resource idle time while the resource is allocated to a task.

In order to extend the applicability of the algorithm to hard real-time applications, we model the jobs of periodic tasks as being split into mandatory and optional parts [7]. Mandatory parts have to be executed completely under all circumstances. Optional parts may be aborted or discarded in case of resource shortage and are thus subject to controllable deadline misses.

Before focusing on algorithmic details and use cases, we present a summary of our design criteria as motivated above:

- The distribution of the execution time should be used instead of just the WCET.
- Utilization efficiency should be reified by reducing reservation times without violating guarantees.
- Mandatory parts of jobs are executed in a hard real-time fashion.
- Optional parts of jobs are executed in a firm real-time fashion with statistical guarantees on deadline misses.
- The fraction of missed deadlines should not only be predictable, but controllable in overload situations.
- The runtime overhead of the scheduling should be minimal.

The first algorithm of this class we developed is Quality-Assuring Scheduling (QAS). The admission algorithm determines the amount of resource to allocate to a task's parts during each period, its so-called reservation time, by consulting the requested quality parameters. The approach is not restricted to a specific resource, the admission model can be adapted to any type of resource.

Scheduling is based on the periodic execution of tasks, with respect to their fixed priorities and reservations. The resource scheduler is responsible for the enforcement of the admitted resource reservation times: a task's optional parts are not allowed to exceed the alloted reservation time. Resources not consumed by any task in a period, i.e., slack times, can be reclaimed to allow further optional parts or best-effort tasks to run, but without providing any guarantees.

The original paper on QAS [10] demonstrates the approach with respect to non-preemptible resources such as disks. The optional parts of a task are divided into a fixed number of disk requests. In this way, the fraction of successfully completely requests represents task quality. Although the approach achieves both 100% resource utilization while satisfying the statistical guarantees, it is restricted to task sets with uniform periods, which limits its applicability. A follow-up technical report [11] overcomes this by extending the admission model to task sets with harmonic and arbitrary periods. However, the admission for arbitrary periods is intractable due to its high complexity.

In this paper, we present a successor to QAS for arbitrary periods, Quality-Rate-Monotonic Scheduling (QRMS). QRMS models resource usage slightly less accurately, but less complex and is therefore practically applicable to a greater degree. It shares parts of admission and scheduling with classical Rate-Monotonic Scheduling (RMS). As such, well-known schedulability test can be applied. The scheduling overhead is as low as that of fixed priority systems.

The remainder of this paper is organized as follows. In the next section, we will summarize the foundations and the task model of our admission controls. Section 3 gives an overview over QAS, followed by an explanation of QRMS in Section 4. The evaluation can be found in Section 5. We discuss related work in Section 6. Section 7 concludes the paper.

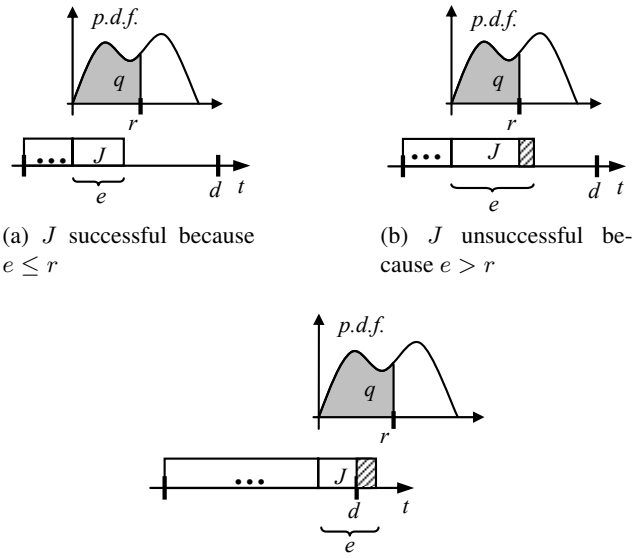
2. Principle Approach

In this section, we explain the basic ideas behind both QAS and QRMS, which are presented in more detail in Sections 3 and 4. Thereafter, we will describe the underlying task model and introduce the notations used throughout the paper.

2.1. The Basic Idea

We aim to achieve sustainably high resource utilization while ensuring a given quality for periodic real-time applications with highly varying resource demands. For that we have developed an admission and scheduling method using fixed priorities and supporting resource reservations. Admission control is based on a probabilistic model which – in contrast to an admission control using WCET – models the actual run-time dispatching in such a system. We use a quality parameter to express the probability that a periodic application meets its deadlines (the percentage of successfully or completely executed jobs of a periodic task). The term *ensured* is used in a probabilistic sense: the longer the task runs the better the match between requested and achieved quality. This approach requires a priori knowledge about the execution time distributions of the tasks.

Our approach realizes the following fundamental idea: Each periodic task may obtain guaranteed use of the processor for a certain constant time span – called reservation time – during each of its periods. If the actual execution time e of a job is shorter than this time span, the remaining time can be used by other tasks, reflecting the usual scheduler behavior to select the next job from the ready queue immediately when a job finishes its execution. If the job would exceed its reservation time, it is aborted. Other than



(a) J successful because $e \leq r$

(b) J unsuccessful because $e > r$

(c) J unsuccessful although $e \leq r$

aborted part of J

Figure 1. Successful and unsuccessful jobs

an admission using WCET, the reservation time is only the fraction of the WCET ensuring a requested quality q. A probabilistic model exactly mapping this scheduling policy enables the computation of these.

The reservation time r appears to result solely from the q-quantile of the execution time distribution of the jobs (see Figures 1a and 1b). However, the model also has to consider a job J being aborted at its relative deadline d even if a complete execution of the job would not have exhausted its reservation time (see Figure 1c).

Hence, the reservation time for each job J of a task requesting a quality q is the shortest time r where

$$\mathbf{P}(J \text{ does not run longer than } r \wedge J \text{ is completed until its relative deadline}) \geq q \quad (1)$$

More formally, let $p_i(r)$ denote the probability that a job of task T_i is completed in the sense of Equation (1) ($r \in \mathbb{R}, i \in \mathbb{N}$). Then we obtain a system of equations for a task set $T = \{T_1, \ldots, T_n\}$ with requested qualities $q_1, \ldots, q_n$:

$$r_i = \min(r \in \mathbb{R} \mid p_i(r) \geq q_i) \quad \forall i = 1, \ldots, n. \quad (2)$$

So the general admission criterion is

$$\text{The system of equations in (2) is solvable.} \quad (3)$$

2.2. Task Model

Generally, we consider tasks T_i as a sequence of jobs J_{ij} to be processed periodically:

$$T_i = (J_{ij})_{j=1,2,\ldots} \qquad i = 1, \ldots, n \quad (4)$$

where $n \in \mathbb{N}$ denotes the total number of tasks in the task set $T = \{T_1, \ldots, T_n\}$.

To be widely applicable, we want to map both hard and firm real-time tasks in both preemptible and non-preemptible flavor to our model. Therefore, each job can be partitioned into one mandatory part M_{ij} and m_i optional parts $O_{ij1}, O_{ij2}, \ldots, O_{ijm}$. M_{ij} is released at the beginning of its respective period, O_{ij1} becomes ready when M_{ij} is completed, and so on. The end of the period is the relative deadline of all parts. The execution times of the parts vary described by random variables, but obviously the mandatory parts do not exceed their WCET, w_i. We assume that the random variables of all tasks are pairwise independent. For a task T_i, the random variables describing the individual instances of the mandatory part are assumed to be identically distributed. The same is assumed for the instances of the optional parts. Finally, an application may specify a minimum percentage, q_i, of optional parts that have to be completed successfully. In summary, the following definition describes a task.

Definition 1 A task T_i is a tuple

$$T_i = (X_i,\ Y_i,\ w_i,\ m_i,\ q_i,\ d_i) \quad (5)$$

where

X_i	nonnegative random variable: execution time of a mandatory part,
Y_i	nonnegative random variable: execution time of an optional part,
w_i	positive real number less than or equal to d_i: worst case execution time of the mandatory parts,
m_i	nonnegative integer: number of optional parts,
q_i	real number $0 \leq q_i \leq 1$: quality parameter, probability that an optional part is completed,
d_i	positive real number: period length = relative deadline.

Further generalizations (task offsets, not identically distributed optional parts) increase the effort to formulate the admission criterion (more variables, more indices), without increasing the tractability or the computational complexity. For simplicity, we identify the parts with their random variables, consider each mandatory part M_{ij} as a realization of X_i and each O_{ijk} as a realization of Y_i. Notably, $m_i = 0$ enables us to model hard real-time tasks, $X_i \equiv 0$ and $m_i = 1$ models a set of "classical" firm tasks (as in [12, 2]).

The admission goal is to derive the priorities $pr(X_i)$ and $pr(Y_i)$ of the mandatory and optional parts and the reservation times r_i from the task description listed above in such a way that a feasible schedule can be generated, by which is meant all mandatory parts meet their deadlines and all optional parts meet their quality requirements. We describe

the approach in three steps: task sets with uniform periods (all tasks have the same length of period d), harmonic periods, and arbitrary periods.

3. Quality-Assuring Scheduling – an Overview

This section outlines the priority assignment and the computation of the reservation times. We conclude it with an evaluation.

3.1. Priority Assignment

In case of uniform periods, we give X_i an arbitrary but high priority because each mandatory part precedes its optional parts and must meet its deadline even in worst-case situations. For the priority assignment of optional parts, we introduced the Quality-Monotonic Scheduling (QMS) in [10] analogous to RMS: a higher quality corresponds to a higher priority. Additionally, the priority of each optional part has to be lower than the priority of each mandatory part. Resulting from empirical investigations, QMS seems to be optimal with respect to feasibility. However, til now a formal proof failed for reasons explained in [11].

We extend QMS for harmonic periods in the following way: a task set T is decomposed into m disjoint subsets $T_1, \ldots, T_m$ such that a subset consists of all tasks with the same period length d_i ($i = 1, \ldots, m$). The subsets are ordered according to period length (T_1 contains the shortest periods and so on). At this point, priorities are assigned according to QMS to both the mandatory and the optional parts of T_1. After that, we treat T_2 the same way but the highest assigned priority must be lower than any priority of T_1, and so on. We use this priority assignment in case of arbitrary periods as well.

3.2. Reservation Times

Due to spatial constraints, we illustrate results for the simplest case only and include some remarks for other cases. We assume uniform periods of length d, optional jobs consisting of one part only, and preemptible resources; furthermore, the task set T has to be ordered according to QMS. Then the reservation time r_i of task T_i given in Equation (2) is calculated as

$$p_i(r) = \mathbf{P}(Y_i \leq r \wedge \sum_{i=1}^{n} X_i + \underbrace{\sum_{j=1}^{i-1} \min(Y_j, r_j)}_{\star} + Y_i \leq d) \quad (6)$$

where the left and right terms of the conjunction respectively corresponds to the abort of an optional part at the end of its reservation time and at the end of its period. The term

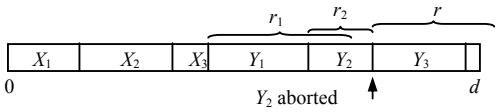


Figure 2. Reservation times of optional parts

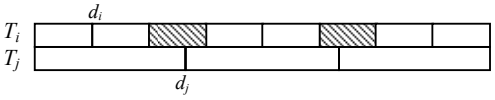


Figure 3. Overlapping periods

$\star$ represents the fact that jobs with higher priorities consume either their actual execution time or at most their reservation time (see Figure 2). Note that the sum of all WCET and all reservation times may exceed the period, nevertheless the resulting schedule is feasible. Due to the existence of mandatory parts, the admission criterion (3) is completed by

$$\sum_{i=1}^{n} w_i \leq d\,. \quad (7)$$

The generalization for harmonic periods is not difficult and is described in [11]. For arbitrary periods, overlapping periods (see Figure 3) pose a problem: Such periods end in a period of the next lower priority task different from the one they began in. The formal model to compute $p_i(r)$ has to consider *all* periods during the hyperperiod of the task set because the available time a job can execute varies from period to period.

Tasks with several optional parts require to generalize Equation (2):

$$r_i = \min(r \in \mathbb{R} \mid \mathbf{E}A_i \geq q_i m_i) \quad (8)$$

$A_i = A_i(r, r_1, \ldots, r_{i-1})$ number of completed optional parts of Task T_i within a period

$\mathbf{E}A_i = \sum_{k=1}^{m_i} k \cdot \mathbf{P}(A_i = k)$ expected value of the random variable A_i

Details can be found in [11, 19]. Finally, the reservation time formula for nonpreemptible resources respects the fact that a started optional part cannot be aborted and so may exceed its reservation time within a period. To avoid pending optional parts at the end of a period, we introduce a WCET $w_{O,i}$ for optional parts of task T_i. For the admission, the period of T_i is diminished by $w_{O,i}$ and the longest WCET of all lower prioritized optional parts. This results in a slightly lower resource utilization but avoids pending optional parts.

3.3. Evaluation

We evaluated the QAS approach with experiments using both a prototype real-time system and simulations for

preemptible (CPU) as well as for nonpreemptible (disk) resources. Furthermore, we also included empirical execution time distributions. Three main conclusions should be emphasized here.

- All the experiments show the compliance of the requested qualities with the achieved qualities.
- The approach enables to provide statistical guarantees and to control the behavior of firm applications even under overload.
- QAS can clearly admit a higher load than an admission based on WCET with negligible loss of quality.

The costs for these advantages are comparatively low. The numerical complexity of the admission control (which can be done offline) is dominated by the convolution of the discretized execution time distributions. The highest complexity is that for the admission in case of nonpreemptible resources; their complexity is $O(s \cdot v^3)$ (s: total number of optional parts, v: common number of values of the random variables) [10, 11]. On the other hand, the scheduler only manages the ready queue based on fixed priorities. So online-overhead is negligible, independent of the type of resources and the type of periods.

In case of arbitrary periods however, the computation of the reservation time is very expensive with increasing costs for larger task sets because the hyperperiod explodes for task sets with close-by period lengths (like 503 and 510) and all periods must be considered. Looking for a way to overcome this difficulty, we propose a new admission control approach, which differs from QAS in three respects: priority assignment, interpretation of the reservation time, and as a consequence, a very low-cost admission algorithm.

4. Quality-Rate-Monotonic Scheduling

We will first explain our new approach, followed by an investigation of the admission performance and overhead.

4.1. The QRMS Approach

QRMS is simple but still effective. We abandon the exact modeling of the scheduling behavior in favor of applying the well-known results from rate-monotonic scheduling theory. Therefore, we choose another priority assignment policy and a simpler way to compute the reservation times:

- Priorities are assigned to *tasks* (this means mandatory and optional parts of a task have the same priority) according to RMS.
- *All* parts of a job are assigned a *common* reservation time.

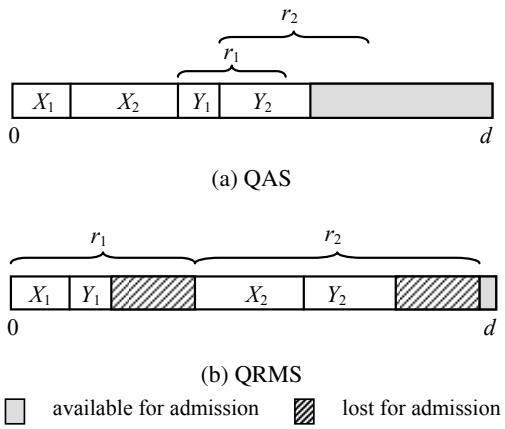


Figure 4. Admission with QAS vs. QRMS

- In the *admission*, the reservation time is regarded as a *constant* execution time.

Consequently, tasks are independent during admission, an important advantage to drastically decrease the admission overhead. Figure 4 illustrates the modified priority assignment and the notion of reservation times for two tasks T_1, T_2 with uniform periods of length d.

The approach uses the task model given in Definition 1. To derive the reservation times, we consider preemptible resources first. We have to use Equation (8). Due to the laws of probability calculus, we can compute the expected value of A_i as

$$\mathbf{E}A_i = \sum_{k=1}^{m_i} \mathbf{P}(A_i \geq k), \quad i = 1, \ldots, n \tag{9}$$

The number A_i of completed parts of task T_i within a period does no longer depend on the reservation times of other tasks. Obviously, it holds (see Figure (5)):

$$\mathbf{P}(A_i(r) \geq k) = \mathbf{P}(X_i + k \cdot Y_i \leq r), \quad k = 1, \ldots, m_i \tag{10}$$

Thus:

$$r'_i = \min(r \in \mathbb{R} \mid \sum_{k=1}^{m_i} \mathbf{P}(X_i + k \cdot Y_i \leq r) \geq q_i m_i)\,. \tag{11}$$

The final formula respects the fact that r'_i may be shorter than the WCET w_i of the mandatory part and includes the constraint that jobs are aborted at the end of their period:

$$r_i = \max(r'_i, w_i) \quad i = 1, \ldots, n \tag{12}$$

We check $r_i \leq d_i$ for all i in a first admission step. Then the final admission test can be done using the Liu/Layland-criterion or time demand analysis [13]. In case of nonpreemptible resources, r'_i is computed as above, but the admission must include the WCET $w_{O,i}$ of an optional part:

$$r_i = \max(r'_i + w_{O,i},\, w_i) \tag{13}$$

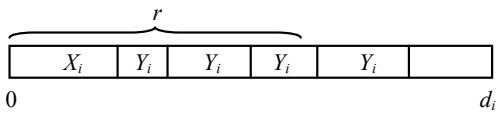


Figure 5. Task with at least two successfully executed optional parts. The third optional part is aborted, the fourth rejected

We call the approach Quality-Rate-Monotonic Scheduling (QRMS). The advantage of QRMS compared to QAS is the simple and low-cost admission even for arbitrary periods. However, by assuming a fixed reservation time in the admission control, QRMS ignores situations where jobs do not completely consume their reservation time (see Figure 4). Thus, QRMS is a more pessimistic admission algorithm compared to QAS (i.e., there are task sets QRMS cannot admit, while QAS can).

4.2. QRMS Admission Performance and Overhead

We will investigate the QRMS admission performance based on quantitative comparisons with other scheduling methods in the next section. Here we elaborate on the observation that QRMS is not optimal. This can be seen easily in the following example. Given two tasks T_i, each with one mandatory part X_i and one optional part Y_i ($i = 1, 2$). All parts are identically uniformly distributed like Z:

Z	1	2
p	0.5	0.5

The task periods are $d_1 = d_2 = 7$, the requested qualities $q_1 = q_2 = 0.9$. The distribution of the sum $X_i + Y_i$

$X_i + Y_i$	2	3	4
p	0.25	0.5	0.25
$\sum p$	0.25	0.75	1

yields the reservation times $r'_1 = r'_2 = 4$, which is greater than the WCET of the mandatory parts, hence $r_1 = r_2 = 4$. Therefore, the admission test fails: $r_1 + r_2 \leq d$ does not hold.

However, the task set is schedulable, because

$$\begin{aligned} \mathbf{P}(X_1 + Y_1 + X_2 + Y_2 \leq 7) &= \\ 1 - \mathbf{P}(X_1 + Y_1 + X_2 + Y_2 = 8) &= \\ 1 - \frac{1}{16} = 0.9275 &> 0.9 \end{aligned} \tag{14}$$

Moreover, the task set can be admitted under QAS. The reservation times are computed for the optional parts only. The 0.9-quantile of Z is 2, obviously, so $r_{1,QAS} = 2$ follows immediately and hence $\min(Y_1, r_1) = Y_1$. Therefore, $r_{2,QAS} = 2$ satisfies Equation (6) because of Equation (14).

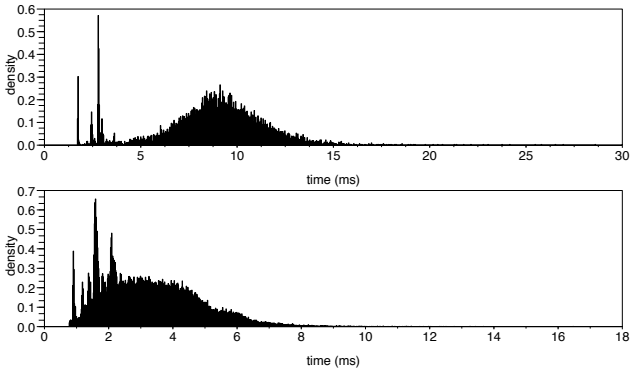


Figure 6. Distributions of Decoding Times of two H.264 Videos, the right edges of the diagrams represent the WCET

As for QAS, the computational complexity of the QRMS admission control is dominated by the required convolutions. Their complexity is $O(s \cdot v^2)$ here, due to the simpler admission. For example, no task set admission considered during the following evaluation took longer than 24 ms on a 2 GHz machine. At runtime, the only overhead of a QRMS implementation is the ready queue management common to all priority-based systems.

5. Evaluation

We have developed a simulation environment which allows to simulate the execution of arbitrary task sets according to our task model. Input parameters are period, quality, the number of optional parts, and instances of the following classes of distributions:

Normal, $N(\mu, \sigma)$	μ: mean value, σ: standard deviation,
Exponential, $E(m)$	m: mean value, i.e., $\alpha = m^{-1}$ is the parameter of the distribution,
Uniform, $U(m, n)$	m: minimum, n: maximum,
$EMP1$ and $EMP2$	two empirical distributions for decoding times of high definition H.264 video (see Figure 6).

The Normal and Exponential Distributions are discretized and truncated at 0 and WCET, if necessary.

To evaluate the performance of QRMS, it was necessary to find algorithms against which QRMS should be compared. However, the search was inconclusive. All the techniques presented in literature provide methods to analyze or to improve a given system's behavior, but fail to allow guiding the system to achieve a requested behavior. SRMS is an exception we found, so it is described and discussed later in

Period	Mandatory Part	WCET	Optional Part	Requested Quality	Reservation Time	Achieved Quality
20	$N(4,1)$	5	$N(3,1)$	**70 %**	7.40	**69.79 %**
20	$N(3,2)$	6	$N(2,1)$	**50 %**	6.00	**69.56 %**
60	$N(6,3)$	10	$N(9,6)$	**75 %**	19.33	**75.50 %**

Table 1. Accuracy of QRMS for Harmonic Periods

Period	Mandatory Part	WCET	Optional Part	Requested Quality	Reservation Time	Achieved Quality
20	$N(5,1)$	6.5	$N(3,1)$	**70 %**	8.58	**70.23 %**
30	$E(0.33)$	4	$N(2,3)$	**90 %**	6.68	**89.72 %**
50	$E(0.25)$	2	$N(5,19.5)$	**80 %**	16.85	**78.44 %**

Table 2. Accuracy of QRMS for Arbitrary Periods

this section. QRMS, providing such control, is discussed in three sets of results: the accuracy of QRMS alone, a comparison to the performance of QAS, and a comparison to RMS.

5.1. Accuracy of QRMS

We conducted exhaustive experiments, of which we present a representative selection. Due to spatial constraints, we restricted the task sets to one optional part. Tables 1 and 2 show the accuracy of QRMS for harmonic and arbitrary periods, respectively. The qualities achieved in the simulation match the requested qualities very accurately. One notable exception is the second task in Table 1, which receives more quality than requested. This is because this task's dominant mandatory part requires worst-case resource reservation according to Equation (12). This results in more optional parts being executed successfully than requested.

Table 3 shows a larger task set, in which we mixed different distributions and included a hard real-time task with no optional part. Table 4 shows a video playback scenario using empirical execution time distributions. The task set consists of optional parts only, but demands high qualities. This allows to schedule three video streams, although the WCETs are large (see Figure 6) compared to the chosen period lengths. Although the resource is fully allocated for all task sets, all achieved qualities tightly match the requested ones.

We want to note that the usual technique of considering large task sets generated at random is of limited value to evaluate QRMS. If a task set is admitted by QRMS, the requested qualities are guaranteed by the proved mathematical foundation. Hence, we demonstrate the accuracy of QRMS with task sets constructed so that any increase in quality or decrease in period leads to a failed admission.

5.2. QRMS versus QAS

When comparing QRMS to QAS, two effects can be observed, both of which are demonstrated on the task set from Table 1. It is possible for QRMS to achieve a higher quality than requested. This happens if the WCET is longer than the reservation time derived from the execution time quantile (see the second task in Table 5). With QAS, the optional parts do not exceed their quality, because of the more complex, but more accurate admission.

The second effect is that task sets exist, which can be admitted by QAS, but not by QRMS. Increasing either the quality of Task 1 from 70 % to 74.5 % or that of Task 3 from 75 % to 94.6 %, the resulting task set is still admitted by QAS, but not by QRMS. A more precise, quantitative analysis of this effect has not been accomplished yet, due to the complexity of the admission formulae and the different meaning of the reservation time, which is applied only to optional parts by QAS, but to mandatory and optional parts combined by QRMS.

5.3. Other Scheduling Policies

First we will refer to the well-known Rate-Monotonic Scheduling, RMS. Obviously, the QRMS admission dominates the RMS admission: Each task set admitted under RMS is schedulable under QRMS with a quality of 100% for each task. Additionally, each overloaded task set with a utilization greater than 1 is schedulable under QRMS given that the qualities are sufficiently small. Thus, we compared qualities ensured by a successful QRMS admission with the deadline miss ratio d_{MR} produced by a corresponding RMS schedule. Table 6 shows the unfairness of RMS: the quality $q = 1 - d_{MR}$ directly depends on the priority, i.e., on the period length.

Statistical Rate Monotonic Scheduling (SRMS) [2] is a generalization of RMS for periodic tasks with highly variable execution times and statistical quality requirements like in QRMS. The idea is to assign an *allowance* similar to our reservation time to a task during its *superperiod,* i.e., the period of the next lower priority task according to RMS. A local admission executed at the release time of every job ensures a percentage of successful jobs to each task. Hence, under SRMS, a released job is rejected im-

Period	Mandatory Part	WCET	Optional Part	Requested Quality	Reservation Time	Achieved Quality
2	$N(0.3, 1)$	0.5	$N(0.5, 0.5)$	**20 %**	0.51	**20.39 %**
6	$N(0.4, 0.4)$	1	$U(1, 2)$	**10 %**	1.43	**10.12 %**
12	$E(0.25)$	1	$E(0.33)$	**90 %**	1.13	**90.23 %**
12	-	-	$U(0, 3)$	**40 %**	1.20	**40.34 %**
36	$E(1)$	2	$EMP1$	**50 %**	3.70	**51.13 %**
36	$N(1, 2)$	3	-	-	3.00	-
72	$E(0.5)$	5	$EMP2$	**70 %**	10.50	**71.46 %**

Table 3. Accuracy of QRMS for a Larger Task Set, one Task has no Mandatory Part, one Task has no Optional Part

Period	Optional Part	Requested Quality	Reservation Time	Achieved Quality
20 ms	$EMP1$	**95 %**	5.90 ms	**95.20 %**
20 ms	$EMP2$	**90 %**	11.80 ms	**90.63 %**
40 ms	$EMP1$	**80 %**	4.50 ms	**81.82 %**

Table 4. Accuracy of QRMS for Empirical Distributions, Tasks consist of Optional Parts only

Req. Quality	QRMS Reservation	QRMS Quality	QAS Reservation (Opt. Part)	QAS Quality
70 %	7.40	69.79 %	3.52	70.12 %
50 %	6.00	**69.56 %**	2.02	**50.06 %**
75 %	19.33	75.50 %	13.41	75.04 %

Table 5. Achieved Qualities of QRMS compared to QAS, Task Set from Table 1

Period	Optional Part	Requested Quality	QRMS Quality	RMS Quality
10	$N(5, 2)$	**20 %**	19.99 %	**99.38 %**
20	$N(10, 4)$	**40 %**	40.01 %	**49.39 %**
50	$N(15, 10)$	**30 %**	28.40 %	**11.00 %**

Table 6. Achieved Qualities of QRMS compared to RMS, Tasks consist of Optional Parts only

mediately, if its complete execution cannot be guaranteed, whereas with QRMS, an aborted job has already consumed resources, which are wasted. Therefore, SRMS performs a priori better than QRMS by achieving higher qualities than QRMS, as can be seen in Table 7. Nevertheless, it is not possible to adequately compare both approaches, as justified in the following section.

6. Related Work

Existing literature offers a large amount of work on

- supporting predictable system behavior in transient or permanent overload situations,
- providing statistical guarantees for soft and firm real-time applications,
- handling execution time variation to improve resource utilization,
- enforcing time restrictions by a reservation based scheduler [1].

To the best of our knowledge, QAS and QRMS are the only approaches which combines all these aspects. In contrast to other methods, they are not restricted to specific resources or applications.

The approach closest to QRMS is SRMS as investigated in Section 5.3. Although it performs generally better than QRMS, it has some practical limitations:

- jobs cannot be divided in parts, so the approach does not cover non-preemptible resources such as disks,
- mandatory jobs could be modeled using a task quality of 100% but WCET cannot be specified,
- the efficiency of the approach depends on the ratio of adjacent periods: it is less efficient for uniform periods and highly expensive for large task sets,
- it causes a non-negligible runtime overhead,

Period	Mandatory Part	WCET	Optional Part	Req. Quality	Reservation	QRMS Quality	SRMS Quality
10	$N(2, 0.5)$	3	$N(1.5, 0.5)$	**70 %**	3.83	70.06 %	**85.9 %**
20	$E(0.33)$	6	$N(2, 1)$	**50 %**	6.00	99.95 %	**77.5 %**
60	$N(6, 3)$	10	$E(10)$	**75 %**	19.56	74.76 %	**79.3 %**

Table 7. QRMS compared to SRMS

- the actual execution time must be known at the release time of each job.

In our opinion, the last limitation makes the approach inapt for real world applications. QRMS avoids this problem by requiring only the execution time distribution, never the exact execution time for a job.

The idea to partition jobs into mandatory and optional parts is taken from the imprecise computation model (ICM) [7]. It handles two types of jobs: N-jobs need not always be executed completely, but the total error is minimized; C-jobs must be completely executed once in a given number of consecutive periods. A class of preemptive, priority-driven scheduling algorithms is proposed but based on WCET, and not applicable for nonpreemptible resources.

The (m,k)-firm tasks model [12] generalizes the idea of C-jobs: a task must meet at least m deadlines within a window of k consecutive invocations. Several papers study schedulability [3, 4, 6] or predictability of missed deadlines [3, 14] for fixed-priority schedulers, [6] provides a response time analysis. Different methods are used to improve the schedulability even under overload:

- partitioning the set of jobs of tasks (not a single job!) into two or three classes similar to mandatory and optional jobs [14, 18], or
- exploiting a history of deadline miss patterns [3, 4, 6] or a history modeled with Markov chains [14], or
- introducing an algebra of temporal constraints to analyze properties of these constraints [3].

An on-line scheduling framework based on two scheduling modes *normal* and *panic* to improve schedulability under overload is presented in [4]. The improvement results from the "inherent pessimism of the [WCET] analysis" ([4]). QRMS avoids this pessimism by using execution time distributions, which are as far as we know used nowhere in (m,k)-firm models. On the other hand, the guarantee provided to a (m,k)-firm task is obviously stronger than the quality in the sense of QRMS: (2,5)-firm is stronger than (20,50)-firm, and both are stronger than 40%.

In a similar way Tia et al. proposed a *transform task method* [20] to provide probabilistic schedulability guarantees to semi-periodic real-time tasks whose ratios of WCET to period length are greater than 1. The authors transform each task into a periodic task followed by a sporadic task, again, similar to mandatory and optional parts. They compute the probability such that each task will meet its deadline, and develop a probabilistic time demand analysis which substitutes the sums of fixed execution times with convolutions of probability density functions. However, the results are not used to control scheduling parameters beforehand. Furthermore, similar to SRMS, the method demands that the exact computation time of each job of a task is known when the task is released.

Diaz et al. [8] describe a stochastic analysis method for a wide class of periodic real-time systems. The proposed method computes the response time distribution of each task based on a Markovian modeling, thus making it possible to determine the deadline miss probability of individual tasks. The computation of the complete probability function of the response time is similar to our approach. Several other papers propose analysis methods for real-time tasks with variable execution times and offer algorithms to compute deadline miss probabilities [1, 5, 9, 15, 16].

Some approaches use statistical methods to improve the utilization of disk drives in multimedia servers. All of these methods aim to calculate the probability of deadline misses for a given workload, based on either a probabilistic model of the disk drive [17, 22] or the measured execution time distribution of disk requests [21].

We notice that related work mostly analyzes the influence of given scheduling parameters on some performance measures such as response time or deadline miss ratio or probability. This only enables to manipulate the parameters in a "trial and error" manner to achieve a desired system behavior. QRMS and QAS calculate the values of the scheduling parameters beforehand to ensure a requested quality or deadline miss ratio.

7. Conclusion

In this paper, we present the Quality-Rate-Monotonic Scheduling (QRMS) admission control and scheduling algorithm. QRMS is the successor to Quality-Assuring Scheduling (QAS), which solved the longstanding problem of efficient resource utilization by exactly modeling how schedulers allocate resources in a real system. QRMS, although less accurate, follows the same idea, but drastically reduces admission complexity compared to QAS. Both these algorithms handle hard as well as firm real-time tasks simultaneously. Hard real-time tasks are represented by mandatory parts, which are always scheduled according to their WCET. Firm tasks, represented by optional parts,

are only guaranteed a fraction of periods in which their corresponding jobs are executed successfully. This fraction is not a result of the admission, it is directly controllable by the application or the user via a quality parameter. Admission control merely guarantees this parameter by deriving a resource reservation time from it. To accomplish that, distribution functions of the jobs' execution times are used rather than pure worst-case times. Such distributions can be obtained empirically. No further knowledge on task execution times is necessary. This sets QAS and QRMS apart from the majority of algorithms in the field: Combining easily controllable probabilistic guarantees, execution time distributions and task partitioning, together with exceptionally low scheduler run-time overhead and the applicability to both preemptible and nonpreemptible resources has to the best of our knowledge not been achieved before with an algorithm as concise and conclusive as QRMS.

References

[1] L. Abbeni and G. Buttazzo. Stochastic Analysis of a Reservation Based System. In *Proc. of the 9th International Workshop on Parallel and Distributed Real-Time Systems*. April 2001.

[2] A. Atlas and A. Bestavros. Statistical Rate Monotonic Scheduling. In *Proc. of the 19th IEEE Real-Time Systems Symposium*. 1998.

[3] G. Bernat, A. Burns, and A. Llamosi. Weakly hard real-time Systems. In *IEEE Transactions on Computers*. 50(4), pp. 308-321, April 2001.

[4] G. Bernat and R. Cayssials. Guaranteed On-line Weakly-Hard Real-Time Systems. In *Proc. of the 22st IEEE International Real-Time Systems Symposium*. pp. 25–37, Dec. 2001.

[5] G. Bernat, A. Colin, and S. Petters. WCET Analysis of Probabilistic Hard Real-Time Systems. In *Proc. of the 23th IEEE Real-Time Systems Symposium (RTSS'02)*. Dec. 2002.

[6] I. Broster, G. Bernat, and A. Burns. Weakly Hard Real-Time Constraints on Controller Area Network. In Proc. 14th Euromicro Conference on Real-Time Systems, pp. 134-141, 2002.

[7] J.-Y. Chung, J. W. S. Liu, and K.-J. Lin. Scheduling Periodic Jobs That Allow Imprecise Results. In *IEEE Transactions on Computers*. vol. 39, no. 9, Sept. 1990.

[8] J. L. Diaz, D. F. Garcia, K. Kim, C.-G. Lee, L. L. Bello, J. M. López, S. L. Min, and O. Mirabella. Stochastic Analysis of Periodic Real-Time Systems. In *Proc. of the 23th IEEE Real-Time Systems Symposium (RTSS'02).* Dec. 2002.

[9] M. K. Gardner and J. W. Liu. Analyzing Stochastic Fixed-Priority Real-Time Systems. In *Proc. of the 5th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. pp. 44–58, March 1999.

[10] Cl.-J. Hamann, J. Löser, L. Reuther, S. Schönberg, J. Wolter, and H. Härtig. Quality-Assuring Scheduling–Using Stochastic Behavior to Improve Resource Utilization. In *Proc. of the 22th Real-Time Systems Symposium (RTSS'01)*. pp. 119–128, December 2001.

[11] Cl.-J. Hamann, L. Reuther, J. Wolter, H. Härtig. Quality-Assuring Scheduling. Technische Universität Dresden, Technical Report, TUD-FI06-09-Dez.2006, December 2006.

[12] M. Hamdaoui and P. Ramanathan. A Dynamic Priority Assignment Technique for Streams with (m,k)-Firm Deadlines. In *IEEE Trans. on Computers*. vol. 44, no. 12, pp. 1443–1451, December 1995.

[13] J. W. S. Liu. Real-Time Systems. Prentice Hall, 2000.

[14] D. Liu, X. Hu, M. Lemmon, and Q. Ling, Firm Real-Time System Scheduling Based on a Novel QoS Constraint. In *Proc. of the 24st IEEE International Real-Time Systems Symposium*, pp. 386–395, Dec. 2003.

[15] S. Manolache, P. Eles, and Z. Peng. Schedulability Analysis of Applications with Stochastic Task Execution Times. In *ACM Journal*. vol. 3, no. 4, Nov. 2004.

[16] A. K. Mok and D. Chen. A Multiframe Model for Real-Time Tasks. In *IEEE Transactions on Software Engineering.* vol. 23, no. 10, pp. 635–645, Oct. 1997.

[17] G. Nerjes, P. Muth, and G. Weikum. Stochastic Service Guarantees for Continuous Data on Multi-Zone Disks. In *Proc. of the 16th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'97)*. pp. 150–160, May 1997.

[18] G. Quan and X. Hu. Enhanced fixed-priority scheduling with (m,k)-firm guarantee. In *Proc. of the 21st IEEE International Real-Time Systems Symposium*, pp. 79–88, Dec. 2000.

[19] L. Reuther. Disk Storage and File Systems with Quality-of-Service Guarantees. Technische Universität Dresden, PhD Thesis, May 2006.

[20] T.-S. Tia, Z. Deng, M. Shankar, M. Storch, J. Sun, L.-C. Wu, and J.-S. Liu, Probabilistic Performance Guarantee for Real-Time Tasks with Varying Computation Times. In *Proc. of the Real-Time Technology and Applications Symposium*. pp. 164–173, May 1995.

[21] H. M. Vin, P. Goyal, and A. Goyal. A Statistical Admission Control Algorithm for Multimedia Servers. In *Proc. of the 2nd ACM International Conference on Multimedia (ACM Multimedia '94)*. pp. 33–40, October 1994.

[22] R. Zimmermann and K. Fu. Comprehensive statistical admission control for streaming media servers. In *Proc. of the 11th ACM International Multimedia Conference (ACM Multimedia 2003)*. pp. 75–85, Nov. 2003.