Performance Evaluation 15 (1992) 203–215
North-Holland

Preemptive LCFS scheduling in hard real-time applications

J. Blieberger and U. Schmid

Department of Automation (183/1), Technical University of Vienna, Treitlstrasse 3/4, A-1040 Vienna, Austria

Received 18 June 1990

Abstract

Blieberger, J. and U. Schmid, Preemptive LCFS scheduling in hard real-time applications, Performance Evaluation 15 (1992) 203–215.

We investigate the real-time behaviour of a (discrete time) single server system with preemptive LCFS task scheduling. The main result deals with the probability distribution of a random variable SRD(T), which describes the time the system operates without violating a fixed task service time deadline T. The tree approach, used for the derivation of our results, is also suitable for revisiting problems in queueing theory.

Relying on a simple general probability model, asymptotic formulas concerning all moments of SRD(T) are determined; for instance, the expectation of SRD(T) is proved to grow exponentially in T, i.e., $E[SRD(T)] \sim C\rho^T T^{3/2}$ for some $\rho > 1$.

Keywords: real-time behaviour, LCFS scheduling, trees, probability generating functions, singularity analysis, asymptotics.

1. Introduction

In this paper, we will study some aspects concerning the real-time behaviour of a discrete single server system with preemptive LCFS task scheduling. Instead of using queueing theory, we apply a special tree approach which is well-known from the analysis of data structures (see [7] or [8] for a survey). A comprehensive discussion of queueing theory may be found in [1] or in [6].

We consider a system containing a task scheduler, a task list of (potential) infinite capacity, and a single server. The scheduler inserts arriving tasks into the task list according to the scheduling strategy. The server always executes the task at the head of the list, thus scheduling is done by rearranging the task list. A dummy task will be generated by the scheduler, if the list becomes empty. If the server executes a dummy task, the system is called *idle*, otherwise *busy*.

Rearranging the task list is assumed to occur at discrete points on the time axis only. The (constant) time interval between two such points is called a *cycle*. Due to this assumption, we are able to model tasks formed by indivisible, i.e., atomic actions with duration of 1 cycle. A task may need an arbitrary number of actions to run to completion, while the dummy task as mentioned above is supposed to consist of a single no-operation action (1 cycle). The *service time* of a task is the time (measured in cycles) from the beginning of the cycle in which the corresponding task is invoked, to the end of the last cycle of that task.

Obviously, the time axis is covered by *idle periods* and *busy periods*, which are supposed to include the initial idle cycle, too. Note that our definitions imply a correspondence between an idle cycle and a busy period with duration of 1 cycle.

Correspondence to: Mr. J. Blieberger, Department of Automation (183/1), Technical University of Vienna, Treitlstrasse 3/4, A-1040 Vienna, Austria.

In order to investigate real-time performance, we are going to study the following random variables:

(1) The *busy period duration* BPD. This is the time interval (measured in cycles) from the beginning of an idle cycle, in which a task arrives, to the end of the last busy cycle induced, i.e., the length of a whole busy period.

We should mention that this duration provides no answer about missing deadlines, since it takes into account the sum of all service times of tasks arriving within the period only, but it should give some insight into system load distributions. BPD is determined by the arrival process and the service times only, hence independent of the scheduling strategy, and has been analyzed by classical queueing theory, too, cf. [6]. Our analysis demonstrates the power of the approach in obtaining the required results quite easily.

(2) The *time to exceed* TTE(T) and the *successful run duration* SRD(T). The former is the time interval (measured in cycles) from the beginning of the initial idle cycle to the beginning of the first cycle causing a fixed task service time deadline of T cycles to be missed, i.e., the time the system operates before a task violates the deadline.

A sequence of nonviolating busy periods followed by a busy period containing at least one deadline violation is called a *run*, the sequence without the last (violating) busy period is referred to as *successful run*. The parameter SRD(T) of the system is the time interval from the beginning of the initial idle cycle to the beginning of the (idle) cycle initiating the busy period containing the first violation of a task's deadline T. Obviously, we have $\mathrm{SRD}(T) < \mathrm{TTE}(T)$.

Since we are concerned with preemptive LCFS scheduling, we can state an exact relation between SRD(T) and TTE(T). As we shall see later, we have $\mathrm{TTE}(T) = \mathrm{SRD}(T) + T$, and it suffices to derive asymptotic results for one of these random variables if the term T is "asymptotically smaller" than the other terms. Thus we will restrict ourselves to studying SRD(T).

We assume an arrival process, which provides an arbitrarily distributed number of task arrivals during a cycle, independent of the arrivals in the preceding cycles, and also of the task execution times. The

Johann Blieberger is currently assistant professor at the Department of Automation at the Technical University of Vienna. His research interests include areas of analysis of algorithms and data structures, theoretical analysis of scheduling algorithms, and real-time programming languages. He studied computer science at the TU Vienna and received his doctorial degree in 1985. From 1984 to 1988 he has been with Siemens AG Austria.

Ulrich Schmid received his diploma (1985) and Dr. techn. degree (1986) in computer science and mathematics from the Technical University of Vienna. Currently, he is assistant professor at the Department of Automation at the TU Vienna. His primary research interests focus on theoretical foundations concerning practical problems within different fields of computer science, especially real-time systems, computer communications and operating systems.

arbitrarily distributed (but independent) task execution time is the number of cycles necessary for processing the task to completion if it occupies the server exclusively.

To make things clearer, we give a few applications of the above. For instance, consider a single processor with a single interrupt line, which executes all machine instructions within a fixed time, a cycle (a few 100 nanoseconds, for example). Traditionally, interrupt arrivals will become recognized at the end of an instruction, causing the CPU to process a (reentrant) service routine. An idle cycle corresponds to the execution of an instruction which is not part of an interrupt service routine.

A straightforward application is the ordinary preemptive LCFS (last come first serve) task scheduling problem for a single processor, though it causes some problems in how to justify an *equidistant* subdivision in atomic actions at a higher level than machine instructions. However, modeling task arrivals by a Poisson process seems to be a reasonable approach.

Another application of the general model may be found in a server for a TDMA channel (time division multiple access). If we consider a single communication channel shared by multiple (say, n) stations, a common approach for synchronizing the transmission activities is TDMA. Each station owns a unique subslot of duration t/n, where it may transmit exclusively (if there is data to transmit, otherwise the subslot is wasted), all together forming a transmission slot of duration t. Due to the cyclic occurrence of the transmission slot, each station may transmit every t time units. A reasonable order of magnitude for t is 10–100 milliseconds.

If we apply our model, transmission slots correspond to cycles and service corresponds to the transmission of one packet. The packet arriving process may be modeled by a Poisson process, an idle cycle corresponds to a wasted (sub)slot.

Notational remarks.
(1) We denote by $[z^n]f(z)$ the nth coefficient in the (formal) power series $f(z)$.
(2) We write $f(x) = O(g(x))$ for $x \to x_0$ if there exists some real constant $C > 0$ independent of x which guarantees $|f(x)| \leqslant C|g(x)|$ for all x in a suitable neighbourhood of x_0.
(3) We use the notation $f(x) \sim g(x)$ for $x \to x_0$ if $\lim_{x \to x_0} f(x)/g(x) = 1$.
(4) We write $f(x) = o(g(x))$ for $x \to x_0$ if $\lim_{x \to x_0} f(x)/g(x) = 0$.

2. Probability model

This section introduces the probability model used for subsequent investigations. We assume arbitrary, but independent probability distributions for the number of task arrivals during a cycle and the task execution times.

The probability generating function (PGF) of the number of task arrivals during a cycle is denoted by

$$A(z) = \sum_{k \geqslant 0} a_k z^k$$

and is constrained by
(1) $A(0) > 0$, i.e., the probability of no arrivals during a cycle is greater than zero. This guarantees the existence of idle periods.
(2) The radius of convergence M of $A(z)$ is greater than 1 since $A^{(r)}(1)$ exists for all $r \geqslant 0$ (cf. [7, p. 98ff]). This implies that all moments of the corresponding random variable are finite.
(3) For purely technical reasons we presume that there exists some σ such that $0 < \sigma < M$ and

$$A(\sigma) = \sigma A'(\sigma).$$

Note that this implies that arrivals during any two different cycles are independent since task arrivals are defined for each cycle separately. It should be noted that all important probability distributions occurring in practical applications are constrained in this way, e.g., the Poisson distribution.

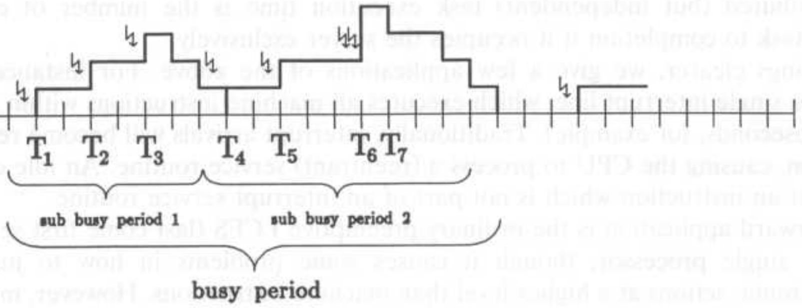

Fig. 1. A simple system with preemptive LCFS scheduling.

The PGF of the task execution time (measured in cycles) is denoted by

$$L(z) = \sum_{k \geq 0} l_k z^k$$

with the following additional assumptions:

(1) $L(0) = 0$, i.e., the task execution time is greater than or equal to one cycle.

(2) The radius of convergence of $L(z)$ is greater than 1. This implies that all moments of the corresponding random variable are finite.

Again, this definition implies task execution times independent of one another and of the arrival process.

3. A tree model

As mentioned in the introduction we are dealing with a discrete time model. To motivate our tree approach for this model, we give an example.

Figure 1 should be interpreted in the following way: Cycles are delimited by short lines on the horizontal axis. Task arrivals are shown by small lightnings. The y-coordinate represents the number of tasks present in the task list. According to our preemptive LCFS scheduling strategy, if only one task arrives during a certain cycle, its execution is started at the next cycle. If more than one task arrives during a cycle, they are executed one after the other, again starting with the next cycle. Executing tasks are represented by horizontal lines. If a task is suspended, because a new task arrives, this is shown by raising the horizontal line of the current task by one unit. If a task terminates, the line lowers one unit. (Note that this is also the case when task T_1 of our example terminates, although task T_4 is started immediately thereafter.) The figure also illustrates the notion of busy and idle periods, which have already been mentioned in the introduction. Another notion, *sub busy periods*, is introduced in the figure. This notion will become important in Section 5.

Now we are coming to a central point of our investigations. We will show how this scheduling model can be represented by *trees*. This kind of notation allows to deduce some results in a quicker and clearer way than, e.g., the Markov chains used in classical queueing theory. For a similar approach see for example [5, p. 300], where a bijection between paths and trees is presented.

In our tree-like notation, a cycle is denoted by a node (○). If k tasks arrive during the processing of a cycle, the corresponding node has k successors, denoted by square nodes (□). According to our probability model, each node with k successors is weighted with a_k, the probability that k tasks arrive during one cycle. If the execution time of a task is n (measured in cycles), the corresponding (square-)node has n successors, being of course ○-nodes. As one would have expected, such a (task-)node is weighted by l_n, the probability of having execution time n.

Thus, our tree consists of two alternating layers, one containing circles (○) and the other containing square nodes (□). For instance, Fig. 1 can be mapped to the following tree (Fig. 2), where weights have

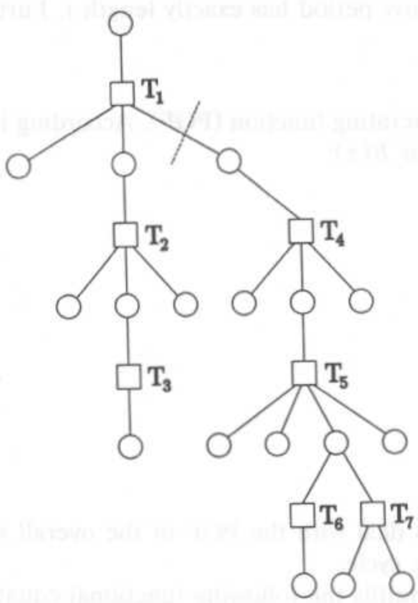

Fig. 2. A simple system modelled by a tree.

been suppressed for the sake of readability. By traversing this tree in preorder, we can reconstruct the original figure as is shown in Fig. 1. The dotted line shows the boundary between two sub busy periods, which will become important in Section 5.

Finally, note that an idle cycle is represented by one ○-node, which may be called a degenerated tree, and that the time axis can be divided into pairs of busy periods and idle periods, where we model a busy period by a (normal) tree and an idle period by a sequence of degenerated trees. Thus the overall system behaviour is represented by a sequence of trees.

4. Busy period duration

In this section we are going to derive several results concerning the length of a busy period. Some of these results give well-known facts from classical queueing theory, which can be deduced more quickly and clearly by using our tree approach. The results of this section will be needed in Section 5, too.

The definition of a busy period can easily be visualized by using *symbolic notation* (cf. [4] or [15]). We denote by $\mathscr{B}$ the family of trees described in the previous section, and by $\mathscr{T}$ the family of (task-)trees:

$$\mathscr{B} = \sum_{k \geqslant 0} a_k \overset{\bigcirc}{\underbrace{\mathscr{T} \ldots \mathscr{T}}_{k\text{ times}}} \qquad \mathscr{T} = \sum_{n \geqslant 1} l_n \overset{\square}{\underbrace{\mathscr{B} \ldots \mathscr{B}}_{n\text{ times}}}$$

As can be seen, these symbolic equations reflect exactly the recursive definition of the trees in Section 3.

The major advantage of this symbolic notation is that it can be translated into functional equations for the corresponding generating functions. For more explanations, see [4] or [15]. The length of a busy period (measured in cycles) is related to the number of nodes of our trees. Thus we can study this quantity by "counting" such trees. More precisely, we do count circles (○) only and not square nodes (□), i.e., the *size* of a tree is given by the number of its ○-nodes. Strictly speaking the term "count" is a little inaccurate; in fact we evaluate the *average size* of trees, where the trees are weighted by the distribution given from $A(z)$ and $L(z)$.

Let b_i be the probability that a busy period has exactly length i. Further let

$$B(z) = \sum_{i \geqslant 1} b_i z^i$$

be the corresponding probability generating function (PGF). According to our symbolic equations we get the following functional equations for $B(z)$:

$$B(z) = \sum_{k \geqslant 0} a_k z T^k(z)$$

and

$$T(z) = \sum_{n \geqslant 1} l_n B^n(z).$$

Thus

$$B(z) = zA(T(z))$$

and

$$T(z) = L(B(z)).$$

Introducing $\phi(z) = A(L(z))$, we can deal with the PGF of the overall service time, i.e., the number of cycles induced by arrivals during one cycle.

Hence we have shown that $B(z)$ fulfils the following functional equation:

Lemma 1. *The probability generating function for the length of a busy period fulfils*

$$B(z) = z\phi(B(z)), \tag{1}$$

where $\phi(t) = A(L(t)) = \sum_{i \geqslant 0} c_i t^i$. *(The PGFs* $A(t)$ *and* $L(t)$ *have been defined in Section 2.)*

The function $\phi(t)$ is analytic in a domain $|t| < R$, $1 < R \leqslant +\infty$ and has non-negative coefficients because of the assumptions on A and L. Let ν (real) be defined by $L(\nu) = \sigma$, where σ is defined in Section 2. (Note that ν is defined uniquely since $L(t)$ is strictly monotonically increasing for $t \geqslant 0$.) Because of the assumptions on A, we have

$$A(L(\nu)) = L(\nu) A'(L(\nu)).$$

Since $L(t) \leqslant tL'(t)$ for $t \geqslant 0$, it follows that there exists some $0 < \tau \leqslant \nu$ such that

$$\phi(\tau) = \tau\phi'(\tau). \tag{2}$$

For practical applications, the deadline T of a task should be large compared to the duration of a cycle. Hence, asymptotic results for large T are satisfactory. In the broad field of *analysis of data structures and algorithms* many tools are used for determining the asymptotic behaviour of coefficients of generating functions. The Darboux/Pólya method is one of them (cf. [2]) and is based on the following theorem:

Theorem (Darboux). *Suppose* $A(z) = \sum_{n \geqslant 0} a_n z^n$ *is analytic near* 0 *and has only algebraic singularities* α_k *on its circle of convergence* $|z| = r$, *i.e., in a neighbourhood of* α_k, *then we have*

$$A(z) \sim \left(1 - \frac{z}{\alpha_k}\right)^{-\omega_k} g_k(z),$$

where $\omega_k \neq 0, -1, -2, \ldots$ *and* $g_k(z)$ *denotes a nonzero analytic function near* α_k. *Let* $\omega = \max_k \Re(\omega_k)$ *denote the maximum of the real part of* ω_k *and by* α_j, ω_j, *and* g_j *denote the values of* α, ω *and* g *such that* $\omega_j = \omega$. *Then we have*

$$a_n = \sum_j \frac{g_j(\alpha_j)}{\Gamma(\omega_j)} n^{\omega_j - 1} \alpha_j^{-n} + o(n^{\omega - 1} r^{-n}).$$

Another well-known tool concerned with functional equations is described in [2].

Theorem (Bender). *Assume that the power series $w(z)=\sum_{n\geqslant 0}a_n z^n$ with nonnegative coefficients satisfies the functional equation $F(z, \omega)\equiv 0$. Suppose that there exist real numbers $r>0$ and $s>a_0$ such that*
(1) for some $\delta>0$, $F(z, w)$ is analytic whenever $|z|<r+\delta$ and $|w|<s+\delta$;
(2) $F(r, s)=F_w(r, s)=0$;
(3) $F_z(r, s)\neq 0$, and $F_{ww}(r, s)\neq 0$; and
(4) if $|z|\leqslant r$, $|w|\leqslant s$ and $F(z, w)=F_w(z, w)=0$, then $z=r$ and $w=s$,
then

$$a_n=\left(\frac{rF_z}{2\pi F_{ww}}\right)^{1/2} n^{-3/2}r^{-n}(1+o(1)),\quad (n\to\infty),$$

where the partial derivatives F_z and F_{ww} are evaluated at $z=r$ and $w=s$.

This theorem is well suited for our needs, so we can derive an asymptotic formula for b_i, the coefficient of z^i in $B(z)$, if we take into account that $B(z)$ satisfies the functional equation (1).

Theorem 1. *Let $d=\gcd\{i\colon c_i\neq 0\}$ denote the number of singularities on the radius of convergence of $B(z)$. The probability for a busy period having length i ($i\equiv 1 \pmod d$) is for $i\to\infty$ given by*

$$b_i=C\rho^{-i}i^{-3/2}(1+o(1)), \tag{3}$$

where $\rho=\tau/\phi(\tau)$, τ is given by (2), and $C=d(\phi(\tau)/2\pi\phi''(\tau))^{1/2}$. (For $i\not\equiv 1$ (d) the probability for a busy period having length i equals 0.)

The factor d results from the fact that $B(z)$ has d singularities on its circle of convergence. Note that $\rho>1$ since $B(z)$ is a probability generating function, and that $\phi(t)=A(L(t))$. Theorem 1 was already obtained in [10] by the Darboux/Pólya method although in the framework of random trees.

It is also possible to determine moments of the parameter BPD directly from the functional equation (1). For instance, we can evalute the mean of BPD

$$\mathrm{E}[\mathrm{BPD}]=B'(1)=\frac{1}{1-\phi'(1)}=\frac{1}{1-A'(1)L'(1)}$$

or the variance

$$\begin{aligned}\mathrm{Var}[\mathrm{BPD}]&=B''(1)+B'(1)-(B'(1))^2\\&=(1-A'(1)L'(1))^{-3}\left[A'(1)L''(1)+A''(1)(L'(1))^2\right]\\&\quad+(1-A'(1)L'(1))^{-2}A'(1)L'(1).\end{aligned}$$

5. The moments of the successful run duration

According to our previous considerations, the time axis can be divided into pairs of busy periods and idle periods. In case of the parameter SRD(T) we need a certain number of such pairs, where the deadline T is not violated during the busy periods, and there is a busy period at the end violating the deadline.

In order to guarantee that a certain deadline T is not violated during a busy period, so-called *sub busy periods* become important. A sub busy period is defined to start either at the beginning of a busy period or when at least one task arrives during the last cycle of the task that started the previous sub busy period. It lasts until the last cycle of the tasks that have started the sub busy period have been processed, but does not include this last cycle. Figure 1 in Section 3 shows that there exists a relation between sub busy periods and those points, where the lines that visualize the tasks, touch the time axis. In fact, sub busy periods are shifted one cycle left w.r.t. the points described above. This shifting has been introduced because it simplifies our computations significantly.

Thus, a busy period is divided into a certain number of sub busy periods (plus one cycle in the end, which we shall consider later). It is clear from the definition of a sub busy period that the deadline T is violated when the length of a sub busy period is greater than T, since we are dealing with preemptive LCFS scheduling. The first deadline violation can only happen to the task having started the sub busy period. Hence, if we are able to find a tree model, we can determine the length of sub busy periods by counting trees.

According to our presumed independent probability distributions, we can model sub busy periods by removing the sub-tree of the last cycle of the rightmost task following the root of our tree $\in \mathscr{B}$, since this is the cycle where the next sub busy period is started (cf. our example in Section 3). Denoting sub busy periods by $\overline{\mathscr{B}}$ and by $\overline{\mathscr{T}}$ in symbolic notation, we get by similar reasoning as in Section 3

$$\overline{\mathscr{B}} = a_0 \bigcirc + \sum_{k \geqslant 1} a_k \overset{\bigcirc}{\underbrace{\mathscr{T} \ldots \mathscr{T}}_{k-1 \text{ times}} \; \overline{\mathscr{T}}}$$

and

$$\overline{\mathscr{T}} = \sum_{n \geqslant 1} l_n \overset{\square}{\underbrace{\mathscr{B} \ldots \mathscr{B}}_{n-1 \text{ times}} \; \Delta}.$$

In the symbolic equation for $\overline{\mathscr{B}}$, the cycle without successors ($a_0 \bigcirc$) is responsible for idle cycles, while the other terms correspond to sub busy periods. Note that we do not count the last cycle of a sub busy period (Δ), since it will be counted correctly in the following sub busy period (or idle cycle).

Thus we get for the corresponding generating functions $\overline{B}(z)$ and $\overline{T}(z)$

$$\overline{B}(z) = \sum_{n \geqslant 1} \overline{b}_n z^n = a_0 z + z \sum_{k \geqslant 1} a_k T(z)^{k-1} \overline{T}(z) = a_0 z + z \frac{\overline{T}(z)}{T(z)} \left(A(T(z)) - a_0 \right)$$

and

$$\overline{T}(z) = \sum_{n \geqslant 1} l_n B(z)^{n-1} = \frac{L(B(z))}{B(z)}.$$

Inserting the formula for $\overline{T}(z)$ into that for $\overline{B}(z)$ we obtain

$$\overline{B}(z) = a_0 z + \frac{z}{B(z)} \left(\phi(B(z)) - a_0 \right),$$

which yields

$$\overline{B}(z) = 1 + a_0 z - \frac{a_0 z}{B(z)} \tag{4}$$

taking (1) into account.

Thus we have found a simple expression for the PGF of the length of sub busy periods. Since, however, we are interested in sub busy periods having length of *at most* T cycles, we have to modify our result. Let $B_T(z)$ be an appropriate GF, viz.

$$B_T(z) = \sum_{n=1}^{T} \overline{b}_n z^n.$$

In order to model the overall system behaviour during a successful run, we obtain for the PGF of the random variable SRD(T)

$$S_T(z) = \sum_{t \geqslant 1} s_t z^t = \frac{1 - B_T(1)}{1 - B_T(z)}.$$

This follows from the fact that the probability generating function of the length of an arbitrary number of non-violating sub busy periods is $\sum_{n\geq 0} B_T(z)^n$, and that the probability of the occurrence of the terminating violation sub busy period equals $1-B_T(1)$.

We are now going to investigate all moments of SRD(T) for large T. So let

$$E^r = E^r(T) = E^r[\mathrm{SRD}(T)]$$

be the rth moment of SRD(T) and let

$$\overline{E}^r = \frac{\mathrm{d}^r}{\mathrm{d}z^r} S_T(z)\bigg|_{z=1} = \sum_{t\geq 1} (t)_r s_t$$

denote the rth factorial moment, where $(t)_r = t(t-1)\cdots(t-r+1)$ is the falling factorial.

Using the formula of Faá di Bruno (cf. [7, p. 50]),

$$\begin{aligned}(b(a(z)))^{(r)}|_{z=t}\\ &= \sum_{j=0}^{r} b^{(j)}(a(t)) \sum_{\substack{k_1+k_2+\cdots+k_r=j\\ k_1+2k_2+\cdots+rk_r=r\\ k_i\geq 0}} \frac{r!}{k_1!(1!)^{k_1}\cdots k_r!(r!)^{k_r}} \left(a^{(1)}(t)\right)^{k_1}\cdots\left(a^{(r)}(t)\right)^{k_r},\end{aligned}$$

it is easy to prove by setting $a(z) = 1 - B_T(z)$ and $b(z) = 1/z$ that

$$\frac{\mathrm{d}^r}{\mathrm{d}z^r} S_T(z) = (1 - B_T(1)) \sum_{j=0}^{r} \frac{(-1)^j j!}{(1-B_T(z))^{j+1}} \mathscr{S}_{j,r}(z),$$

where $\mathscr{S}_{j,r}$ denotes terms involving derivatives of $B_T(z)$ according to the formula of Faá di Bruno.

Hence we get for $\overline{E}^r$

$$\overline{E}^r = \sum_{j=0}^{r} \frac{(-1)^j j!}{(1-B_T(1))^j} \mathscr{S}_{j,r}(1).$$

Next we investigate $B_T(1)$. Since we will also need asymptotic information on $B_T^{(r)}(1) = \mathrm{d}^r/\mathrm{d}z^r B_T(z)|_{z=1}$, we are going to deal with this more general case in the following lemma.

Lemma 2. *With the notations above, we have for $T\to\infty$*

$$B_T^{(r)}(1) = \overline{B}^{(r)}(1) + \frac{(T)_r \overline{b}_T \rho^r}{1-\rho}(1+o(1)),$$

where $\overline{B}^{(r)}(z)$ denotes the rth derivative of $\overline{B}(z)$ and $(T)_r = T(T-1)\cdots(T-r+1)$.

Proof. We define the generating function $H^{(r)}(z)$ by

$$\begin{aligned} H^{(r)}(z) &= \sum_{T\geq 1} B_T^{(r)}(1) z^T = \sum_{T\geq 1}\left(\sum_{n=1}^{T} (n)_r \overline{b}_n\right) z^T \\ &= \sum_{n\geq 1} (n)_r \overline{b}_n \sum_{T\geq n} z^T = \sum_{n\geq 1} (n)_r \overline{b}_n \frac{z^n}{1-z} = \frac{z^r}{1-z} \overline{B}^{(r)}(z). \end{aligned}$$

Thus we have

$$B_T^{(r)}(1) = [z^T] \frac{z^r \overline{B}^{(r)}(z)}{1-z}.$$

Since we want to apply the Darboux/Pólya method again, we have to determine the singularities of $H^{(r)}(z)$. In order to determine these singularities, we first note that $B(\xi) = 0$ can only happen if $\xi = 0$.

This can be seen by regarding (1) which implies $0 = a_0\xi$ if we set $z = \xi$. Hence (4) does not cause a singularity at $z = 0$ and there remain two possible types of singularities:
(1) the pole at $z = 1$, and
(2) the singularities of $\bar{B}^{(r)}(z)$.
The singularity of $\bar{B}^{(r)}(z)$ nearest to the origin is equal to the singularity of $B(z)$ nearest to the origin (cf. (4)) and is thus known to be greater than 1 (cf. (3)). Thus we consider the pole at $z = 1$ first.

Let

$$R^{(r)}(z) = H^{(r)}(z) - \frac{\bar{B}^{(r)}(1)}{1-z}.$$

Now, the function $R^{(r)}(z)$ does not have a pole at $z = 1$, which can be checked using de l'Hospital's rule. Thus

$$B_T^{(r)}(1) = \bar{B}^{(r)}(1) + [z^T]R^{(r)}(z).$$

On the other hand, $R^{(r)}(z)$ still has a singularity at $z = \rho$ originating from $\bar{B}^{(r)}(z)$. Clearly $z = \rho$ is now the singularity of $R^{(r)}(z)$ nearest to the origin. Hence

$$B_T^{(r)}(1) = \bar{B}^{(r)}(1) + \frac{(T)_r \bar{b}_T \rho^r}{1-\rho}(1 + o(1)), \ (T \to \infty)$$

by expanding $H^{(r)}(z)$ around $z = \rho$. □

Note that for $r = 0$ we obtain

$$B_T(1) = 1 + \frac{\bar{b}_T}{1-\rho}(1 + o(1)), \quad (T \to \infty).$$

Taking into account (3) and Faá di Bruno's formula, we see that the major contribution for the asymptotic behaviour of $\bar{E}^r$ comes from the term where $j = r$, because the inner sum of Faá di Bruno's formula contributes $\mathscr{S}_{j,r}(1) = O(1)$ for $T \to \infty$ (cf. Lemma 2). The other terms of the outer sum ($j \neq r$) can be neglected, since their denominators are of smaller order.

Thus we get for $r \geqslant 1$ the following asymptotic expression

$$\bar{E}^r(T) \sim r! \left[\frac{B_T'(1)}{1 - B_T(1)} \right]^r, \quad (T \to \infty)$$

because $\mathscr{S}_{r,r}(1) = (-1)^r (B_T'(1))^r$. From this it follows that for $T \to \infty$

$$E^r(T) \sim \bar{E}^r(T),$$

i.e., the rth moment is asymptotically equivalent to the rth factorial moment.

To derive a more illustrative asymptotic formula for the rth moment, we let

$$\mu = \mu(T) = E^1(T) = \frac{B_T'(1)}{1 - B_T(1)},$$

and investigate $\mu(T)$ for $T \to \infty$ in the following treatment. Observing Lemma 2, we get

$$B_T'(1) = a_0 B'(1) + \mathrm{O}(\rho^{-T} T^{-1/2}).$$

In order to find an asymptotic expansion for $\bar{b}_T$, which is needed in studying $1 - B_T(1)$ for $T \to \infty$, we are going to determine the asymptotic behaviour of $\bar{B}(z)$ near its singularity ρ. From (1) and the theorems due to Darboux and Bender cited in Section 4 we conclude that $B(z)$ has the following expansion near its singularity ρ:

$$B(z) = \tau - c_1(1 - z/\rho)^{1/2} + \mathrm{O}(1 - z/\rho),$$

where $c_1 = (2\phi(\tau)/\phi''(\tau))^{1/2}$. Inserting this into (4) yields for $z \to \rho$

$$\begin{aligned}\overline{B}(z) &= 1 + a_0\rho - a_0\rho\left(\tau - c_1(1 - z/\rho)^{1/2} + \mathrm{O}(1 - z/\rho)\right)^{-1} \\ &= 1 + a_0\rho - \frac{a_0\rho}{\tau}\left(1 + \frac{c_1}{\tau}(1 - z/\rho)^{1/2} + \mathrm{O}(1 - z/\rho)\right)\end{aligned}$$

Using Darboux's Theorem again, this implies

$$\overline{b}_T \sim \frac{a_0\rho}{\tau^2} b_T \quad \text{as } T \to \infty.$$

Before we state our main result, we have to reconsider our system modeling. Remember that we did not take into account the last cycle of a sub busy period since it is considered to be the starting cycle of the following period. So we have computed SRD($T+1$), and not SRD(T). We fix this by decrementing T by one in our derivations and obtain the following main result.

Theorem 2. *With the notations above, as $T \to \infty$, the rth moment of* SRD(T) *fulfils*

$$E^r(T) \sim r!\mu(T)^r,$$

where

$$\mu(T) = \frac{B_T'(1)}{1 - B_T(1)}$$

and

$$\mu(T) = \frac{1}{d}\left(\frac{2\pi\phi''(\tau)}{\phi(\tau)}\right)^{1/2} \frac{\tau^2(\rho - 1)}{\rho^2(1 - \phi'(1))}\rho^T T^{3/2}(1 + \mathrm{o}(1)).$$

Thus we see that the mean of SRD(T) grows exponentially in T. Note that the moments of the random variable SRD(T) are asymptotically equivalent to the moments of an exponential distribution with parameter $1/\mu(T)$. Unfortunately, we are not able to derive the limiting distribution of the random variable SRD(T). This is due to the asymptotic nature of our results which are not uniformly valid for $r \geqslant 1$.

We conclude this section by relating the two random variables SRD(T) and TTE(T) which have been described in the introduction.

Since the task which violated the given deadline T first, is a task that started a sub busy period, we have the following simple relation between SRD(T) and TTE(T):

$$\mathrm{TTE}(T) = \mathrm{SRD}(T) + T.$$

Hence, the asymptotic formulas derived for the moments of SRD(T) are valid for the random variable TTE(T), too.

6. Poisson arrivals

This section deals with an application of the preceding general formulas to the Poisson case. We consider Poisson-type arrivals and a fixed task execution time of one cycle. Let λ (as usual) be the parameter of the Poisson distribution, then we get

$$\phi(z) = \mathrm{e}^{\lambda(z-1)}.$$

Applying Theorem 1, the PGF of the length of a busy period $B(z)$ fulfils

$$B(z) = \mathrm{e}^{\lambda(B(z)-1)}, \tag{5}$$

and we get for the probability of a busy period having length i

$$b_i = C\rho^{-i} i^{-3/2}(1+o(1)) \quad \text{for } i \to \infty, \tag{6}$$

where $\rho = e^{\lambda-1}/\lambda$ and $C = (\lambda\sqrt{2\pi})^{-1}$.

Fortunately in the case of Poisson distributed arrivals one can find an exact formula for b_i by applying the *Lagrange inversion formula* to (5) (cf. e.g. [15]). We get

$$b_i = e^{-\lambda i}\frac{(\lambda i)^{i-1}}{i!},$$

but we will not need this for our further considerations. Needless to say, our asymptotic formula can be derived from the exact formula by applying Stirling's approximation to $i!$.

For the mean of BPD we obtain in this special case

$$E[\text{BPD}] = B'(1) = \frac{1}{1-\lambda}$$

and for the variance

$$\text{Var}[\text{BPD}] = B''(1) + B'(1) - (B'(1))^2 = \frac{\lambda}{(1-\lambda)^3}.$$

Both of these formulas are well-known from classical queueing theory, cf. [6].

Results concerning the parameter SRD(T) for Poisson arrivals can be derived using our Theorem 2. As $T \to \infty$, we find for the rth moment of this random variable

$$E^r[\text{SRD}(T)] \sim r!\mu(T)^r,$$

where

$$\mu(T) \sim \sqrt{2\pi}\,\frac{\lambda\, e^{2(1-\lambda)}}{1-\lambda}(\rho-1)\rho^T T^{3/2}.$$

7. Conclusions

We would like to remark that this paper should not be seen as a major contribution to *combinatorics*. Rather, to the authors' knowledge it is the first theoretical result covering topics of *hard real-time systems*, and its aim is to present our tree approach as a new method in obtaining results that could not be derived using classical queueing theory.

However, it should be noted that results concerning the maxima of data structures, such as lists and queues, have already been obtained by several authors with the help of probabilistic tools (cf. e.g. [9,11,14]). These results have important applications to preallocating resources, but are not applicable to hard real-time systems, where the size of data structures is less important than the capability of the system to respond to certain events within a given time constraint.

We would also like to mention that our tree approach can be extended easily to other scheduling algorithms. For example we were able to study *FCFS scheduling* using a similar, but more complicated tree model [12]. Comparing these results with those of this paper shows that FCFS scheduling should be preferred in a hard real-time system, if our probability model can be applied.

Under the impression of these results one could have expected that it is the preemptive property of LCFS scheduling that causes it to behave worse than FCFS, because it is always a task starting a sub busy period that violates the given deadline. But we proved that this is wrong by studying *non-preemptive LCFS scheduling* [13], where it is shown that this scheduling algorithm behaves very similar to its preemptive counterpart.

It should be noted that the authors have also studied FCFS scheduling under *rush-hour conditions* [3], i.e., under conditions where the system is not able to cope with the arriving tasks.

References

[1] A.O. Allen, *Probability, Statistics, and Queueing Theory* (Academic Press, Orlando, FL, 1978).

[2] E.A. Bender, Asymptotic methods in enumeration, *SIAM Rev.* **16** (1974) 485–515.

[3] J. Blieberger and U. Schmid, FCFS scheduling in a hard real-time environment under rush-hour conditions, *BIT* (1991) to appear.

[4] P. Flajolet, Analyse d'algorithmes de manipulation d'arbes et de fichiers, Thèse Paris-Sud-Orsay, 1979; *Cahiers de BURO* **34–35** (1981) 1–209.

[5] I.P. Goulden and D.M. Jackson, *Combinatorial Enumeration* (Wiley, New York, NY, 1983).

[6] L. Kleinrock, *Queueing Systems, Vols. 1 and 2* (Wiley, New York, NY, 1975).

[7] D.E. Knuth, *The Art of Computer Programming, Vol. 1: Fundamental Algorithms* (Addison-Wesley, Reading, MA, 1973, 2nd edn.).

[8] D.E. Knuth, *The Art of Computer Programming, Vol. 3: Sorting and Searching* (Addison-Wesley, Reading, MA, 1973).

[9] C.M. Kenyon-Mathieu and J.S. Vitter, General methods for the analysis of the maximum size of data structures, in: Proc. 16th Annual International Colloquium on Automata, Languages, and Programming (ICALP 1989), Stresa, Italy, July 1989, *Lectures Notes in Computer Science, Vol. 226* (Springer, Berlin, 1989).

[10] A. Meir and J.W. Moon, On the altitude of nodes in random trees, *Canad. J. Math.* **30** (1978) 997–1015.

[11] C.M. Mathieu and J.S. Vitter, Maximum queue size and hashing with lazy deletion, in: Proc. 20th Annual Symposium on the Interface of Computing Science and Statistics, Reston, VA, April 1988, pp. 743–748.

[12] U. Schmid and J. Blieberger, Some investigations on FCFS scheduling in hard real-time applications, *J. Comput. System Sci.* (1991) to appear.

[13] U. Schmid and J. Blieberger, On non-preemptive LCFS scheduling with deadlines, 1991, to appear.

[14] L. Takács, *Combinatorial Methods in the Theory of Stochastic Processes* (Robert E. Krieger Publishing Company, Huntington, NY, 1977).

[15] J.S. Vitter and P. Flajolet, Average-case analysis of algorithms and data structures, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. A: Algorithms and Complexity* (North-Holland, Amsterdam, 1990) 431–524.