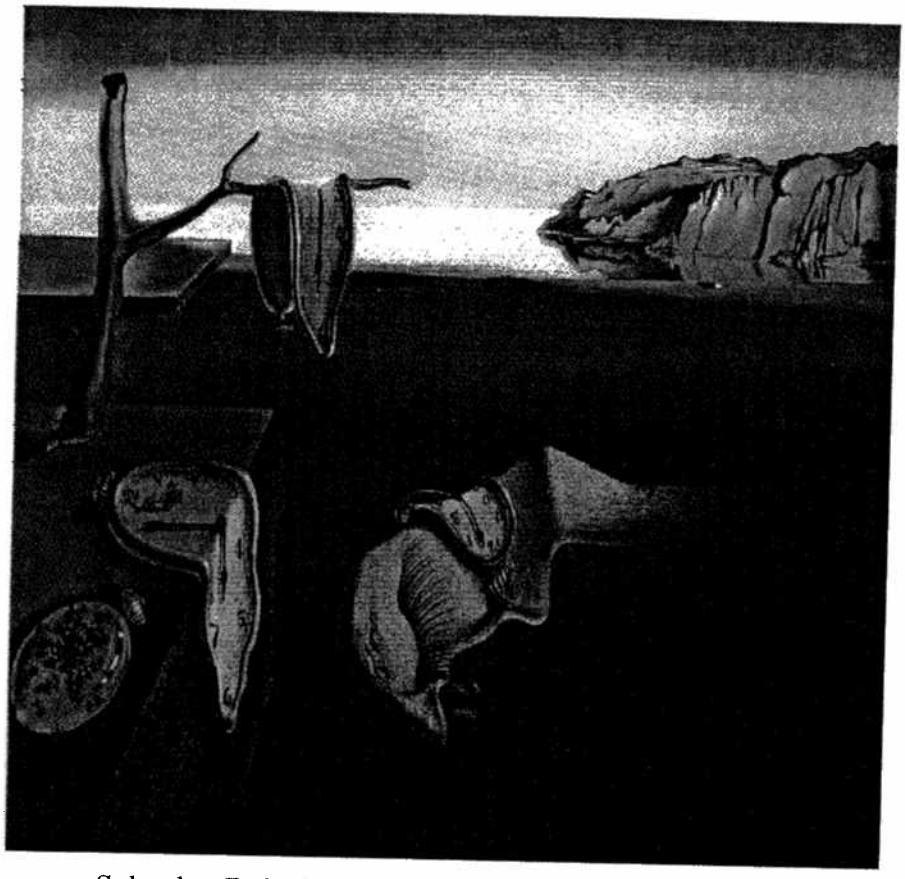
Institut für Automation
Abt. für Automatisierungssysteme

Technische
Universität
Wien

Projektbericht Nr. 183/1-60
August 1995

Random Trees in Queueing Systems with Deadlines (Habilitationsschrift)

Ulrich Schmid



Salvador Dali, "Die Beständigkeit der Erinnerung"

Random Trees in Queueing Systems with Deadlines

Habilitationsschrift

Eingereicht an der

Technisch-Naturwissenschaftlichen Fakultät

der

Technischen Universität Wien

von

Ulrich Schmid
Technische Universität Wien
Institut für Automation
Treitlstraße 3, A-1040 Wien

◇

Meinen Lehrern gewidmet

◇

10. Juni 1994

Theoretical Computer Science (to appear)

Random Trees in Queueing Systems with Deadlines

U. SCHMID[1]

Worinne einem anfahenden Organisten
Anleitung gegeben wird, auff allerhand Arth
einen Choral durchzuführen, anbey auch
sich im Pedalstudio zu habilitieren, indem
in solchen darinne befindlichen Chorälen
das Pedal gantz obligat tractieret wird.
Preface of the "Orgelbüchlein" of Johann Sebastian Bach.

Abstract. We survey our research on scheduling aperiodic tasks in real-time systems in order to illustrate the benefits of modelling queueing systems by means of random trees. Relying on a discrete-time single-server queueing system, we investigated deadline meeting properties of several scheduling algorithms employed for servicing probabilistically arriving tasks, characterized by arbitrary arrival and execution time distributions and a constant service time deadline T. Taking a non-queueing theory approach (i.e., without stable-state assumptions), we found that the probability distribution of the random time S_T where such a system operates without violating any task's deadline is approximately exponential with parameter $\lambda_T = 1/\mu_T$, with the expectation $E[S_T] = \mu_T$ growing exponentially in T. The value μ_T depends on the particular scheduling algorithm, and its derivation is based on the combinatorial and asymptotic analysis of certain random trees. This paper demonstrates that random trees provide an efficient common framework to deal with different scheduling disciplines and gives an overview of the various combinatorial and asymptotic methods used in the appropriate analysis.

1. INTRODUCTION

Scheduling has always been one of the key issues in computer science. In almost any computing system there are concurrent activities competing for mutually exclusive resources, and if there are not sufficiently many resources available or if a single shared resource is to be used, a schedule assigning activities to resources over time is needed. Research on scheduling —which owes much to the research on machine scheduling and related problems in operations research conducted several decades ago, see [DLR82], for example— is traditionally performed in the context of processor scheduling, see e.g. [Kle75]. Note, however, that shared resources like communication channels are becoming more and more important given the trend towards parallel and distributed systems.

Real-time systems, on the other hand, are a relatively young but increasingly important branch of computer industries. Spacecrafts, power plants, automated factories, but also various multimedia applications are examples of such systems. Generally speaking, tasks of a real-time system have to be performed not only in a correct, but also in a timely

[1]Department of Automation (183/1), Technical University of Vienna, Treitlstraße 3, A-1040 Vienna. Email: s@auto.tuwien.ac.at

fashion. Usually they must finish within a predefined deadline[2], otherwise there might be more (*hard real-time*) or less (*soft real-time*) severe consequences.

Scheduling goals for real-time systems are obviously different from those fitting the needs of ordinary computer systems. In fact, it is not hard to show that timeliness is not a simple consequence of high throughput or similar performance characteristics, see [**TK91**], [**CSR88**] for an introduction and overview.

The problem is sufficiently well-understood for *deterministic* tasks, in particular for *periodic* ones as introduced by polling techniques in safety-critical hard real-time systems. Since (future) task arrivals are fully deterministic, schedules may even be determined in advance, i.e., *offline*, see [**BES93**] for a thorough overview. Systems relying on such assumptions are usually called *static*, and their most attractive property is the possibility of (*a priori*) *guarantees* of timeliness.

Static systems, however, are of limited applicability and somewhat unflexible, so that *dynamic systems* are common in practice. Task arrivals in dynamic systems may be *aperiodic*, i.e., arbitrary, and are therefore not known in advance (there is no *clairvoyancy*), so scheduling must be performed *online*. This, of course, rules out a number of computationally expensive (offline) algorithms to be used online, but the most distinctive property w.r.t. to static systems is the possibility of (transient) overloads. As a consequence, no (unrestricted) *a priori* guarantee of timeliness may be given any more[3], see e.g. [**CSB90**], [**ZS89**]. Moreover, the behaviour of online scheduling algorithms under overload is often totally different from the non-overloaded one. Note also that there is a principal deficiency w.r.t. deterministic (i.e., clairvoyant) algorithms under overload conditions, see [**BKM91**], [**KS92**].

Of course, uncertainities in aperiodic task arrivals call for probabilistic modelling, and queueing theory has indeed been applied to related problems in operations research for decades. However, apart from the question whether existing real-time systems are adequately modelled by usual queueing system assumptions, there is also the problem of drawing meaningful conclusions on the actual operation of a real-time system from steady-state results like waiting time distributions or the percentage of task losses/rejections (we will briefly return to this question in the following Section 2). Therefore, a sound theoretical framework for scheduling in dynamic (soft) real-time systems is lacking.

Some of our research is devoted to this problem domain. More specifically, our aim aim is to quantify deadline meeting properties of scheduling algorithms for probabilistic aperiodic tasks in real-time systems. Based on a simple discrete-time queueing system model, we found a suitable —and mathematically tractable— quality measure which has been successfully applied to compare a number of different scheduling algorithms. The complete derivation of our major results is contained in a number of papers published elsewhere, cf. [**BS92**], [**SB92**], [**BS91**], [**SB94**], [**S94**], [**DS93**]. The goal of this paper is

[2]Somewhat different real-time requirements are to be met in B-ISDN (broadband integrated service and data networks) supporting multi-media applications. There is an increasing interest among the research community in establishing reasonable qualities of service (QOS), see [**Tow93**], [**Kur93**] for details. That research, however, is not within the scope of our paper.

[3]Note however that there are ideas like [**LR92**] how to integrate a dynamic system into a static one, preserving guarantees for the static part of the system.

to emphasize the power of modelling scheduling disciplines in queueing systems by means of random trees. Our approach unifies (and simplifies) the investigation of several different queueing problems by utilizing powerful combinatorial and asymptotic methods from the analysis of algorithms and data structures, thus providing a promising alternative to the usual queueing theory devices.

The outline of our paper is as follows: Section 2 contains a description of the underlying model and a very brief survey of related (queueing theory) approaches, Section 3 provides the definition of the quantities of interest and some general preliminaries. Sections 4, 5, and 6 are devoted to the investigation of *preemptive last come first served* (LCFS), *first come first served* (FCFS), and *nonpreemptive last come first served* scheduling for the simple no-priority case, Section 7 surveys the analysis of the important *static priority scheduling algorithm*. Finally, some conclusions and directions of further research are appended in Section 8.

2. The Model

Our investigations are based on a discrete time queueing system consisting of a task scheduler, a task list of (potential) infinite capacity, and a single server. Arriving tasks are inserted into the task list by the scheduler according to the particular scheduling algorithm. A dummy task is generated by the scheduler if the list becomes empty. The server always executes the task at the head of the list, so that scheduling is done by rearranging the entries of the task list. If the server executes a dummy task, the system is called *idle*, otherwise *busy*.

Rearranging of the task list (= scheduling) occurs only at discrete points in time, without any overhead. The length of the interval between two such points is an integral multiple of some unit time called a (machine) *cycle*. Due to this assumption, we are able to model tasks formed by non-preemptible *actions* with duration of 1 cycle. The *task execution time* of a task is the number of cycles necessary for processing the task to completion if it occupies the server exclusively. An ordinary task may have an arbitrary task execution time, a dummy task as mentioned above consists 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 task arrives at the system to the end of the cycle which completes the execution of the task.

Before we proceed, two applications of our model are given. First, consider a single processor with an interrupt line, which executes all machine instructions within a fixed time, a cycle. Usually interrupt arrivals become recognized at the end of an instruction, causing the CPU to process a certain service routine. An idle cycle corresponds to the execution of an instruction not part of an interrupt service routine.

Another example may be found in a (single) client of a TDMA (time division multiple access) channel. If a communication channel is to be shared by multiple (say, n) stations, a common approach for synchronizing the transmission activities is TDMA. Each client owns a dedicated subslot of duration t/n, where it may transmit exclusively if there are data to transmit, otherwise the subslot gets wasted. All subslots together form a transmission slot (cycle) of duration t. Due to the cyclic occurrence of the transmission slot, each client may transmit every t time units. To apply our model, we let a cycle be equal to the length of

a transmission slot and assume a "task execution time" of one cycle, that is, "service" is actually the transmission of a packet. An idle cycle corresponds to a wasted transmission slot.

For our (input-)probability model, we assume arbitrarily distributed task arrivals within a cycle, independent of the arrivals of the preceding cycles, and independent of the arbitrarily distributed task execution times as well. The *service time deadline* of a task is assumed to be constant (fixed).

The *probability generating function* (PGF) of the number of task[4] arrivals during a cycle is denoted by

$$A(z) = \sum_{k \geq 0} a_k z^k, \qquad \text{where } a_k = \text{prob}\{k \text{ tasks arrive during a cycle}\} \tag{2.1}$$

and should meet the constraint $a_0 > 0$, assuring the existence of idle cycles. The PGF of task execution times (measured in cycles) is denoted by

$$L(z) = \sum_{k \geq 1} l_k z^k, \qquad \text{where } l_k = \text{prob}\{\text{task execution time is } k \text{ cycles}\} \tag{2.2}$$

with the additional assumption $L(0) = 0$. It turns out that the overall execution time, i.e., the number of cycles necessary for processing all actions induced by task arrivals during one cycle, plays a central role. The corresponding PGF evaluates to

$$P(z) = \sum_{n \geq 0} p_n z^n = A\big(L(z)\big). \tag{2.3}$$

For the sake of simplicity, we omit the discussion of some necessary "technical" conditions on $A(z)$, $L(z)$ and $P(z)$. Most of them are analyticity requirements which are usually easy to establish. Note however, that we are explicitely excluding the trivial case $P(z) = p_0 + (1 - p_0)z$.

We should mention that the number of *globally* valid arrival distributions meeting our constraints is considerably limited due to the required independency. Globally valid distributions consistent with our assumptions must be based on an interarrival distribution with the memoryless property, i.e., an exponential or geometric distribution, leading to (well-thumbed) Poisson- or Bernoulli-type arrivals within a cycle. In terms of queueing theory, we are therefore dealing with a M/G/1 system[5], which has of course been extensively studied. We will conclude this section by briefly relating our results to that research; a comprehensive and in-depth treatment of single-server queues may be found in Cohen's book [**Coh82**].

[4] We introduce the no-priority case here; generalizing to multiple PGFs arising in the case of several priority levels should be straightforward.

[5] For the remainder of this section, we will adhere to the usual queueing theory terminology: Discussing an M/G/1 system means that we are considering customers arriving at a single server (1), with exponentially distributed interarrival times (M for Markovian) and arbitrarily distributed service times (G for general).

Classical queueing systems are based on continuous time and assume service in the order of arrivals (FCFS). Among the quantities (random variables) of interest are the *queue size*, the *waiting time* (in queue) or *sojourn time* (in queue + service), and the *server utilization*, for example. Basically, there are two different results: (1) *time-dependent solutions* describing, say, queue size at some time t (starting from some initial state at t=0), and (2) *steady-state results* obtained by letting $t \to \infty$. Steady-state results are particularily attractive, since they lead to relatively simple expressions, something that is by far not true for time-dependent solutions. However, one should bear in mind that steady-state results are reasonably meaningful only because most queueing systems "converge" to a stable equilibrium, which is independent of the initial state *if they are left to themselves for a sufficiently long time.*

Therefore, most work on queueing systems is devoted to steady-state results. In particular, for M/G/1 queues, waiting time distributions are available for numerous scheduling disciplines, including random order, inverse order of arrival (LCFS), priority queueing with static or dynamic priorities, etc., see [**Kle75**, Vol. 2, Chap. 3]. The same is obviously true for most of the results obtained by queueing theory in the real-time context.

For instance, there is a well-studied class of queues with *impatient customers*, which are of some interest for dynamic real-time systems. The basic idea is to impose a bound on the waiting ([**BBH84**]) or sojourn time ([**GS77**]) and force customers to leave the system if that bound is (or, alternatively, will be) exceeded. Stable-state results for quantities like waiting time distribution, customer rejection probability, etc. for various scheduling disciplines are available, cf. the nice overwiew in [**ZS89**].

Much effort has also been spent on developing scheduling disciplines that are optimal in various respects. Optimality research goes back to operations research problems as machine scheduling, aiming at scheduling algorithms that minimize various cost functions like the expected number of late jobs, see e.g. [**P83**]. More recent research in the real-time systems area deals with algorithms minimizing the maximum lateness ([**SG92**]) or the fraction of customers exceeding their deadlines ([**PTW88**]), for example.

Although steady-state results give some insight in long-term operation of a real-time scheduling algorithm, they are useless to characterize short-term operation. Steady-state results are obviously incapable of capturing transient phenomena —like (first) entrance into some particular state— that ultimately determine the (time-dependent) statistics of the maximum of a random variable. Such results, however, are the only ones that are really meaningful for determining short-term deadline meeting capabilities when hard real-time requirements are present.

Attacking this type of problems requires advanced mathematical methods. For example, in [**Coh82**] a first entrance time approach —similar to our one— is applied to an FCFS queueing system with bounded waiting time (impatient customers) to derive results on the maximum of certain random variables (Chap. III.4.1, III.7.4). In particular, an integral representation of the Laplace transform of the maximum waiting time during the busy period of a G/G/1 FCFS queueing system is provided (Chap. III.4.1). This formula might be used as an immediate starting point to derive a continuous-time analogon to our result in Section 5.

A similar approach is used in [**LKS91**] to derive asymptotic results on the maximum

queue size for a number of queueing disciplines. Note that such results also emanated from the analysis of hashing with lazy deletion (HwLD), see [**KV91**], [**AHS92**]. As an alternative, large deviation methods can be used to compute first entrance times in Markov-processes, see [**MS93**], [**M91**] for only two applications. We do not know of any particular work in the real-time systems area, but we are convinced that this powerful technique might be successfully applied in our context as well.

Finally, in [**Ta77**] an interesting combinatorial approach towards the maximum of a certain sum of random variables is developed, which is based on an extension of the classical ballot-problem. It is applicable to a wide variety of stochastic processes, including the M/G/1 FCFS system, thereby providing an elegant way of deriving maximum waiting time distributions and related quantities. Note that some quantities found in our analysis in Section 5 appear in [**Ta77**] also; in fact, it should be possible to derive our FCFS-result by means of that approach as well.

Although there are possibly a few alternatives to our analysis of FCFS scheduling, we do not know of any work that deals with the other scheduling disciplines successfully solved by our method. Actually, we think that all the abovementioned approaches are at least difficult to apply, since the waiting time process does not have such a simple description for disciplines other than FCFS. Moreover, we are interested in sojourn times and not in waiting times, and it is the discrete time model and not the continuous one that is really suitable for our problem domain.

3. The Successful Run Duration

Our basic idea concerning a pertinent quality criterion for scheduling algorithms in real-time systems was to consider quantities related to the time that passes until the very first violation of a deadline, but we had to recognize soon that approaching this quantity directly was difficult. The following alternative, however, was found to be successful[6]: The operation of our system may be viewed as a sequence of successive bulks of busy cycles, seperated by one or more idle cycles. Consequently, we define a *busy period* as an initial idle cycle and all busy cycles induced by task arrivals during this busy period. For instance, an (initial) idle cycle with no task arrivals forms a trivial busy period with duration of 1 cycle. By virtue of this definition (and our probability model), the operation of our system may be modelled as a sequence of mutually independent busy periods $\{\mathcal{B}^{(i)};\ i = 1, 2, \dots\}$.

We call a busy period *T-feasible*, if all tasks serviced during the busy period meet their *service time deadline* T. In addition, a sequence of T-feasible busy periods followed by a non-feasible busy period (containing at least one deadline violation) is called a *T-run*, the sequence without the terminating non-feasible busy period is refered as a *successful T-run*. The random variable *successful T-run duration* $\mathcal{S}_T$, which denotes the time interval from the beginning of an (initial) busy period to the beginning of the (idle) cycle initiating the busy period containing the first violation of a task's deadline T, was found to be a suitable[7] mathematically tractable quality criterion.

[6] By using a very similar approach, we also solved the old problem of analyzing the duration of the successful operation of the well-known slotted ALOHA collision resolution algorithm, which we found exponentially distributed too; see [**DS93b**] and [**Drm91**] for details.

[7] In fact, in any case investigated so far, it is not hard to prove that the difference between $\mathcal{S}_T$ and the time up to the actual violation is (asymptotically) negligible.

The probability distribution of $\mathcal{S}_T$ (even its expectation) allows to compare the performance of different scheduling algorithms. Apart from comparison, it also provides an answer to the following practical question: Given the input probability distributions for a certain (high-)load situation, and a (tolerable) probability p for deadline missing (say, $p = 10^{-9}$), what is the maximum duration such a situation could last in order to guarantee a deadline missing probability of at most p? Since μ_T increases exponentially with the deadline(s), such systems may be expected to operate properly a long time (at least in case of reasonable input conditions); this ultimatively explains why they work reasonably well in practice.

Since the (feasible) busy periods constituting a successful run are mutually independent, it is in fact easy to evaluate the PGF of $\mathcal{S}_T$ by means of the PGF of a T-feasible busy period. Let $b_{k,T} = \text{prob}\{\text{length of a } T\text{-feasible busy period is } k \text{ cycles}\}$ and

$$B_T(z) = \sum_{k \geq 0} b_{k,T} z^k \tag{3.1}$$

be the corresponding (improper, that is $B_T(1) < 1$) PGF. Then, the PGF of the random variable $\mathcal{S}_T$ is given by

$$S_T(z) = \sum_{k \geq 0} s_{k,T} z^k = \frac{1 - B_T(1)}{1 - B_T(z)}, \tag{3.2}$$

where of course $s_{k,T} = \text{prob}\{\text{length of a successful } T\text{-run is } k \text{ cycles}\}$. This follows easily from the fact that the PGF of the length of an arbitrary number of T-feasible busy periods is $\sum_{n \geq 0} B_T(z)^n$, and that the probability of the occurrence of the terminating non-feasible busy period equals $1 - B_T(1)$.

Thus, we can reduce the investigation of different scheduling techniques to the analysis of T-feasible busy periods, i.e., the evaluation of $B_T(z)$. For example, the expectation of $\mathcal{S}_T$ yields

$$\mathrm{E}[\mathcal{S}_T] = \mu_T = S'(1) = \frac{B'_T(1)}{1 - B_T(1)}. \tag{3.3}$$

Hence, all what is needed for the expectation are asymptotic expressions for $B_T(1)$ and $B'_T(1)$ for $T \to \infty$, that is, the first few terms of the Taylor expansion of $B_T(z)$ at $z = 1$.

Fortunately, much more can be said about the distribution of $\mathcal{S}_T$. It turns out that it is necessary to distinguish three different situations:

(1) *Normal Case*

This (most important) case is characterized by an average offered load of less than 100%, which may be expressed by $P'(1) < 1$, since $P'(1)$ equals the average number of actions caused by task arrivals within a cycle, cf. equation (2.3). In other words, our system has to deal with task arrivals keeping it not totally busy on the average.

A careful treatment of equation (3.2) reveals a surprisingly simple general statement concerning the probability distribution of $\mathcal{S}_T$ in this particular case. As we have shown in [**DS93**], $\mathcal{S}_T$ is approximately exponentially distributed with parameter $\lambda_T = 1/\mu_T$.

We reformulate two of the most important theorems of [**DS93**]. First, by means of singularity analysis techniques on $S_T(z)$, we obtained asymptotic expressions for the distribution function $\sum_{k=0}^{n} s_{k,T}$ when $n \to \infty$ and $T \to \infty$:

THEOREM 3.1. (Asymptotics of $\sum s_{k,T}$, cf. [**DS93**, Theorem 3.5]) *There exists some $\delta > 0$ such that the distribution function $v_{n,T} = \sum_{k=0}^{n} s_{k,T}$ of $\mathcal{S}_T$ has a uniform asymptotic expansion*

$$v_{n,T} = 1 - \big(1 + O(1/\mu_T)\big) e^{-\mu_T^{-1}\left(1+O(1/\mu_T)\right)n} + O\big(\mu_T^{-1}(1+\delta)^{-n}\big)$$

for $n \to \infty$ and $T \to \infty$.

Second, using Mellin transform techniques, we derived uniform asymptotic expansions for the m-th moment $\mathrm{E}[\mathcal{S}_T^m]$ of $\mathcal{S}_T$:

THEOREM 3.2. (Asymptotics of the Moments of $\mathcal{S}_T$, cf. [**DS93**, Theorem 3.7]) *There exists some $\delta > 0$ such that the moments $\mathrm{E}[\mathcal{S}_T^m]$ of $\mathcal{S}_T$ have the uniform asymptotic expansion*

$$\mathrm{E}[\mathcal{S}_T^m] = \sum_{n \geq 1} n^m s_{n,T} = m! \big[\mu_T\big(1 + O(1/\mu_T)\big)\big]^m + O\left(\mu_T^{-1} \frac{m!\sqrt{m}}{(2\pi e \delta)^m}\right)$$

for $T \to \infty$ and $m \geq 1$.

(2) *Balanced Case*
Here our system is kept 100% busy on the average, i.e., $P'(1) = 1$.
Unfortunately, our convenient Theorems 3.1 and 3.2 are no longer valid in the balanced case since the limiting PGF $\lim_{T\to\infty} B_T(z)$ has radius of convergence $R = 1$, violating condition $R > 1$ of [**DS93**]. Hence, we restricted ourselves to the computation of the first few moments of $\mathcal{S}_T$ in this less important case.

(3) *Overloaded Case*
This case may be characterized by an average offered load which is higher than the maximum load the system is able to cope with, formally, $P'(1) > 1$.
Our theorems do not apply in this case either. The reason is that $\lim_{T\to\infty} B_T(z)$ is no longer an ordinary PGF, but rather an improper one. Thus, $\lim_{T\to\infty} B_T(1) < 1$, which violates another precondition of [**DS93**]. Unlike the two cases above there is a non-zero probability of the occurrence of a busy period that never terminates! Again, we had to confine ourselves with the computation of the first few moments of $\mathcal{S}_T$.

In any case, the problem of determining the distribution of $\mathcal{S}_T$ boils down to the investigation of $B_T(z)$; the properties of a particular scheduling algorithm are "contained" in $B_T(z)$ and are carried over to $S_T(z)$ by equation (3.2). Obviously, different PGFs $B_T(z)$ are obtained for different scheduling algorithms.

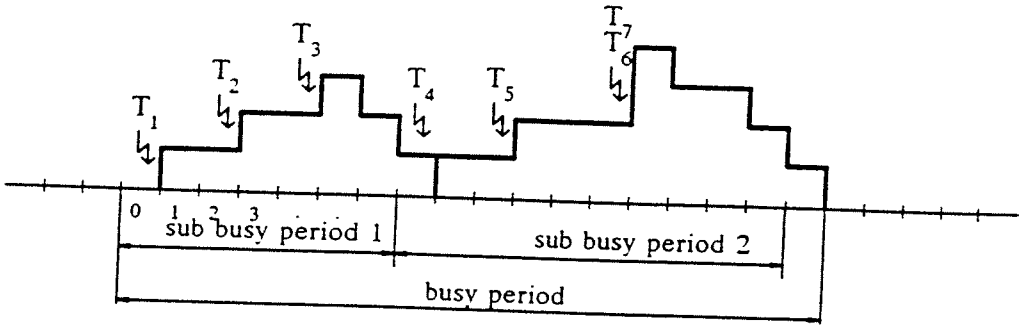
The basic idea underlying our treatment of $B_T(z)$ is to establish a one-to-one correspondence between T-feasible busy periods and a certain family of random trees. Since the probability weights of random trees have the same compositional properties as counting

weights of ordinary trees (the probability of the union and intersection of two disjoint and independent events equals the sum and the product, respectively, as it is the case for cardinalities of sets), the whole theory of translating admissible combinatorial constructions expressible via *symbolic equations* to the corresponding *ordinary generating functions* (OGF) apply; see [**Fla79**] or [**VF**] for details. This usually provides a functional equation for the OGF.

Since the OGF of the family of random trees corresponding to T-feasible busy periods is of course exactly the required PGF $B_T(z)$ (because of the probability weights), that combinatorial translations provides us immediately with a functional equation for $B_T(z)$. Easy-to-use expressions for the required values $B_T(1)$ and $B'_T(1)$ for large T are eventually determined by means of more or less straightforward asymptotic methods applied to (the functional equation for) $B_T(z)$.

4. Preemptive LCFS Scheduling

This section deals with the investigation of T-feasible busy periods for (no-priority) preemptive LCFS (last come first served) scheduling worked out in [**BS92**]. The algorithm is very simple: If a task arrives during a cycle, its execution is started at the beginning of the next cycle, preempting the currently executing task. If there are several tasks arriving during the same cycle, they are executed one after the other, in some arbitrary —preferably LCFS— sequence. Consider the following example:

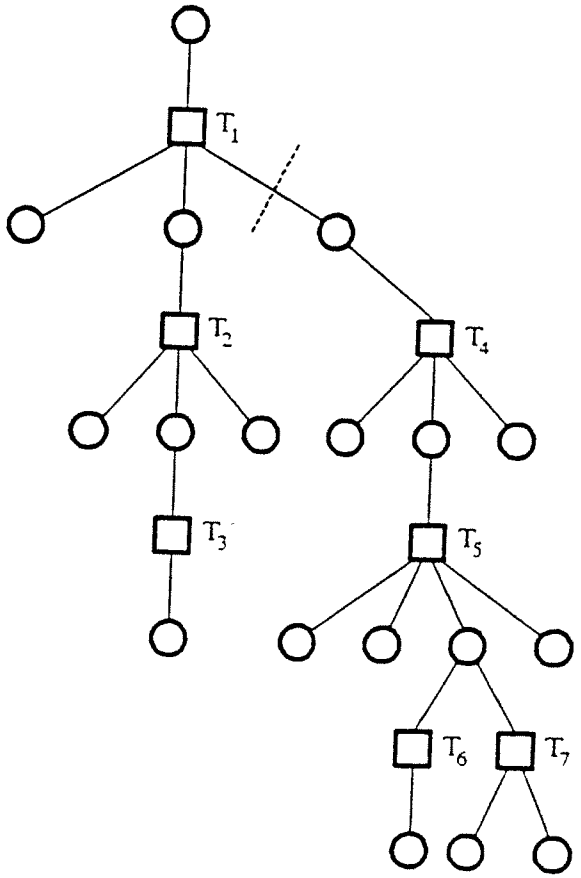


The horizontal axis represents the time-axis with one cycle per division; those cycles forming the busy period of interest are numbered consecutively. Task arrivals are shown by small lightnings with task-names above, executing tasks are represented by horizontal lines. The vertical level of a line represents the number of preempted tasks (plus 1).

In our example above there is a point where a task (T_1) finishes and another task (T_4) starts its execution immediately thereafter. Such situations are closely related to *sub busy periods*: A sub busy period is defined to start either at the beginning of a busy period, or at the beginning of the last cycle of the task that initiated the previous sub busy period, provided that at least one task arrives during that cycle. A sub busy period terminates at the beginning of the next sub busy period. Thus, a busy period may be viewed as the concatenation of an arbitrary number of sub busy periods plus one cycle at the end. Our figure shows the relation between sub busy periods and those points touching the horizontal axis mentioned above: sub busy periods are shifted one cycle left w.r.t. those points. This shifting has been introduced because it simplifies our computations considerably.

It is easy to verify that, as far as deadline missing is concerned, the task initiating a sub busy period is badly off: Any subsequent task arrival causes a preemption of this task until the execution of all the newcomers is complete. Thus, if the length of a sub busy period (which is equal to the service time of the task initiating the sub busy period minus 1) is less or equal to $T-1$ cycles, it is guaranteed that all tasks processed during the sub busy period have a service time of less or equal T cycles, i.e., meet their deadlines. Conversely, if a task with a service time greater than T cycles is processed during a sub busy period, the initiating task experiences a deadline violation as well.

Our major step is the construction of a certain family of random trees that correspond to our busy periods, and the restriction to a sub-family covering exactly T-feasible busy periods. This is done in the following way: A single cycle is denoted by a circular node ◯. Such a node has n successors denoted by square nodes □, if exactly n tasks arrive during the corresponding cycle, and is weighted with the appropriate probability a_n. A square node corresponds to a task and has k ◯-successors if the task execution time of the task is k cycles. It is weighted by l_k, the probability of a task execution time of k cycles. Thus, our tree consists of two alternating layers, one containing circular ◯ and the other square nodes □ only. The following tree corresponds to the figure above (weights have been suppressed):



The original figure is reconstructed by traversing this tree in preorder. Note the dotted line, which marks the boundary between two consecutive sub busy periods.

Letting aside feasible busy periods for the moment, we first look at arbitrary busy periods. The appropriate results are of course well-known in queueing theory, but it seems remarkable how simple they follow from our tree approach. Denoting by $\mathcal{B}$ the family of trees with a circular root, and by $\mathcal{T}$ the family of (task-)trees with a square root, we obtain the following symbolic equations:

$$\mathcal{B} = \sum_{n\geq 0} a_n \quad \underbrace{\mathcal{T} \quad \cdots \quad \mathcal{T}}_{n \text{ times}}$$

and

$$\mathcal{T} = \sum_{k\geq 1} l_k \quad \underbrace{\mathcal{B} \quad \cdots \quad \mathcal{B}}_{k \text{ times}} \tag{4.1}$$

According to [**VF90**], we just have to mark each circular node with the counting variable z and to apply straightforward product and sum translations to obtain the OGFs

$$\begin{aligned} B(z) &= z \sum_{n\geq 0} a_n T(z)^n \\ T(z) &= \sum_{k\geq 1} l_k B(z)^k \end{aligned} \tag{4.2}$$

and eventually

$$B(z) = zP\big(B(z)\big), \tag{4.3}$$

remember equation (2.3). Note that we count circular nodes only, and not square nodes, since the "size" of a tree is the number of its ○-nodes. $B(z)$ is of course exactly the PGF of (unrestricted) busy periods, that is, if b_k denotes the probability that a busy period $\mathcal{B}$ has length k, we have

$$B(z) = \sum_{k\geq 1} b_k z^k,$$

remember our remarks at the end of Section 3.

Starting from equation (4.3) it is easy to obtain classical queueing theory results, e.g., the expectation of $\mathcal{B}$ evaluates to

$$\mathrm{E}[\mathcal{B}] = B'(1) = \frac{1}{1 - P'(1)}.$$

Moreover, by applying standard asymptotic techniques (cf. [**Ben74**] and [MM78], for example) to equation (4.3) it is possible to obtain an asymptotic expansion for $B(z)$ near

its dominant (algebraic) singularity $\rho = \tau/P(\tau)$, τ denoting the (unique) positive solution of $P(x) = xP'(x)$:

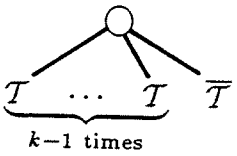
$$B(z) = \tau - b \cdot (1 - z/\rho)^{1/2} + O(1 - z/\rho) \qquad \text{for } z \to \rho, \tag{4.4}$$

where $b = \sqrt{2P(\tau)/P''(\tau)}$. This leads to an asymptotic expression for b_n, namely

$$b_n = \frac{b}{2\sqrt{\pi}} \rho^{-n} n^{-3/2} \left(1 + O(1/n)\right) \qquad \text{for } n \to \infty. \tag{4.5}$$

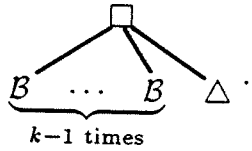
Now we will return to the problem of investigating T-feasible busy periods. We have already mentioned that deadline missing is intimately related to the length of sub busy periods: a deadline T is violated iff the length of a sub busy period is greater than $T - 1$. Hence, we have to consider a special sub-family $\overline{\mathcal{B}}$ of our family of trees $\mathcal{B}$ first, which represents sub busy periods. The symbolic equations are easily found:

$$\overline{\mathcal{B}} = a_0 \bigcirc + \sum_{k \geq 1} a_k$$



and

$$\overline{\mathcal{T}} = \sum_{k \geq 1} l_k$$



The cycle without successors ($a_0\bigcirc$) in the symbolic equation for $\overline{\mathcal{B}}$ is responsible for idle cycles (trivial sub busy periods). Since we do not count the last cycle of a non-trivial sub busy period ($\triangle$), it is possible to paste a number of non-trivial sub busy periods together; the last cycle of a sub busy period is counted correctly in the following sub busy period.

The corresponding generating functions $\overline{B}(z)$ and $\overline{T}(z)$ evaluate to

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

and

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

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

$$\overline{B}(z) = 1 + a_0 z - \frac{a_0 z}{B(z)}. \tag{4.6}$$

Now, the improper PGF for T-feasible sub busy periods (not exceeding a length of $T-1$ cycles) is

$$\overline{B}_T(z) = \sum_{n=1}^{T-1} \overline{b}_n z^n, \tag{4.7}$$

and a T-feasible busy period is formed by the concatenation of an arbitrary number of non-trivial T-feasible sub busy periods terminated by a single idle cycle. The improper PGF yields

$$B_T(z) = \sum_{k\geq 0} \left(\overline{B}_T(z) - a_0 z\right)^k a_0 z = \frac{a_0 z}{1 + a_0 z - \overline{B}_T(z)}. \tag{4.8}$$

What remains to be done is the computation of asymptotic expansions for $B_T(1)$ and $B'_T(1)$ as $T \to \infty$, cf. equation (3.3). Equation (4.8) shows that this requires expressions for $\overline{B}_T(1)$ and $\overline{B}'_T(1)$. However, remembering (4.7), the latter are easily obtained by applying straightforward singularity analysis techniques to the generating function

$$H_r(z) = \sum_{T\geq 1} \overline{B}^{(r)}_{T+1}(1) z^T = \frac{z^r}{1-z} \overline{B}^{(r)}(z), \tag{4.9}$$

where we need to consider $r = 0$ and $r = 1$ only.

In the normal case, we note a simple pole at $z = 1$ and an algebraic singularity at $z = \rho > 1$, remember (4.6) and (4.4). It hence follows almost immediately that

$$\overline{B}^{(r)}_T(1) = \overline{B}^{(r)}(1) + \frac{a_0 \rho^{r+1} (T)_r b_{T-1}}{\tau^2(1-\rho)} \left(1 + O(1/T)\right) \qquad \text{for } T \to \infty, \tag{4.10}$$

with $(T)_r = T(T-1)\cdots(T-r+1)$. After some straightforward algebra, we obtain our major result:

THEOREM 4.1. (preemptive LCFS scheduling in the normal case, cf. [**BS92**, Theorem 2]). *The successful T-run duration $\mathcal{S}_T$ for preemptive LCFS scheduling in the normal case is approximately exponentially distributed with parameter $1/\mu_T^{pLCFS}$, where*

$$\mu_T^{pLCFS} = \left(\frac{2\pi P''(\tau)}{P(\tau)}\right)^{1/2} \frac{\tau^2(\rho-1)}{\rho^2\left(1-P'(1)\right)} T^{3/2} \rho^T \left(1 + O(1/T)\right) \qquad \text{for } T \to \infty,$$

$\tau > 1$ is the solution of $P(x) = xP'(x)$, and $\rho = \tau/P(\tau) > 1$. ∎

In the balanced case, it turns out that $B(z)$ has radius of convergence $\rho = 1$. However, a refined analysis of Equation (4.9) for $H_r(z)$ yields an asymptotic expression for $\overline{B}^{(r)}_T(1)$ even in this case. We just have to take into account that the simple pole at $z = 1$ and the algebraic singularity at $z = \rho = 1$ join for an algebraic singularity of appropriate order. Following the same line of derivation as above, we obtain

THEOREM 4.2. (preemptive LCFS scheduling in the balanced case). *The expectation of the successful T-run duration S_T for preemptive LCFS scheduling in the balanced case is given by*

$$\mu_T^{pLCFS^b} \sim T \qquad \text{for } T \to \infty. \quad \blacksquare$$

In the overloaded case, expansion (4.10) is in fact valid again since $\rho > 1$ (though $\tau < 1$). However, since $B(1) = \beta < 1$ and hence $\overline{B}(1) < 1$ by (4.6), it suffices to plug the coarser expression $\overline{B}_T^{(r)}(1) = \overline{B}^{(r)}(1) + o(1)$ into Equation (4.8) (and its derivative) to arrive at the appropriate

THEOREM 4.3. (preemptive LCFS scheduling in the overloaded case). *The expectation of the successful T-run duration S_T for preemptive LCFS scheduling in the overloaded case is given by*

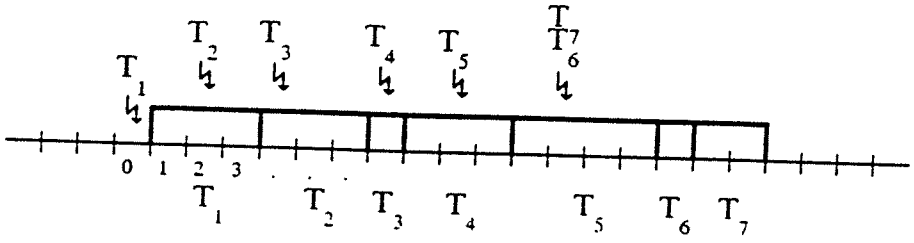
$$\mu_T^{pLCFS^o} \sim \frac{\beta}{1-\beta} \cdot \frac{1}{1-P'(\beta)} \qquad \text{for } T \to \infty,$$

where $\beta < 1$ is the solution of $x = P(x)$, $x < 1$. $\blacksquare$

5. FCFS SCHEDULING

This section is devoted to the investigation of T-feasible busy periods for (no-priority) FCFS (first come first served) scheduling contained in [**SB92**]. Note that, due to our fixed deadline assumption, FCFS scheduling is in fact equivalent to the *earliest deadline first* algorithm here. Tasks are simply executed in the order of arrival, so it makes sense to consider an *expanded task list* which contains all the actions the tasks in the original task list consist of; note that we assume that the actual execution time of a task is determined at the time the task arrives at the system.

Consider the following example, which is based on the task arrivals used in Section 4:



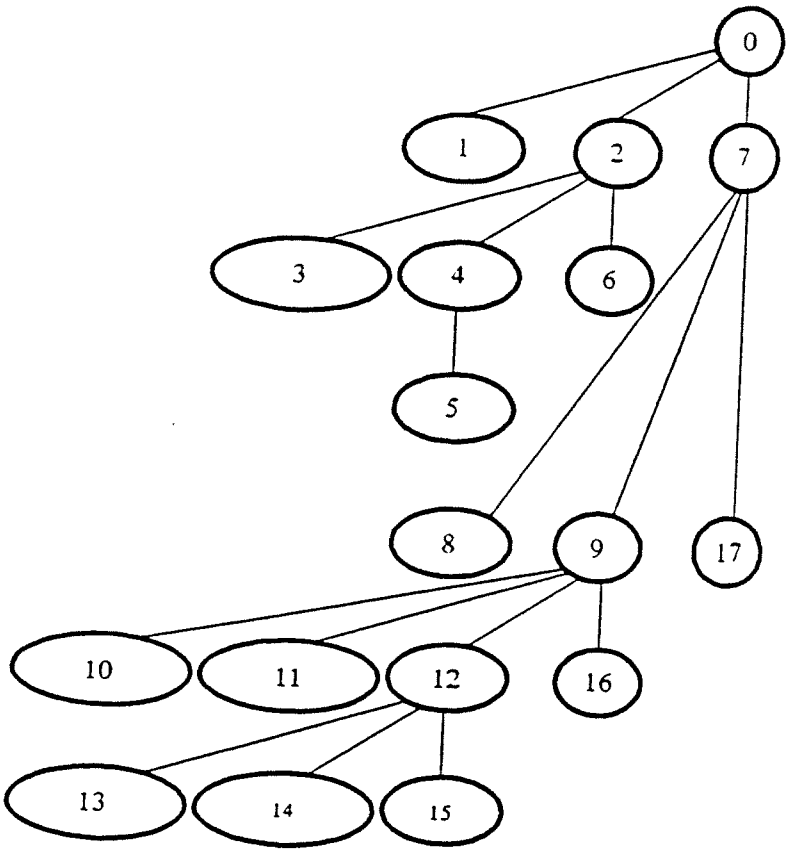
This figure has to be interpreted like the one in Section 4. Note that the execution of each task is represented by a horizontal line (whose length obviously equals the task execution time) at one and the same level, since there is no preemption in FCFS scheduling. For the sake of readability, we attached the name of the corresponding task to each such line.

We will again establish a one-to-one mapping between busy periods and certain random trees, which provides a nice correspondence between feasible busy periods and trees of limited "width". Note that this family of trees appears in the analysis of a simple register function for T-ary operations, too; cf. [**KP87**], [**FRV79**] for details.

Our construction is as follows: Each vertex corresponds to a single cycle; the root of the tree represents the initial idle cycle (number 0). A vertex has n successors if the sum

of the task execution times of all tasks arriving during the corresponding cycle is n; it is weighted by the appropriate probability p_n, cf. equation (2.3). Moreover, each vertex is also labeled by the number of the corresponding cycle within the busy period. This labeling is obtained by a preorder traversal (left to right) of the tree (which also allows to reconstruct the original busy period from the tree).

The following tree corresponds to the busy period above:

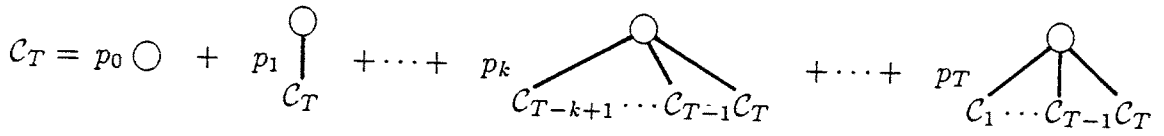


Let us, for example, consider the node with label 1: At the beginning of the corresponding cycle 1 in the busy period, the initial (idle) action has just left the task list and the first action of the task that arrived during the initial cycle is to be executed. One encounters that, due to our special alignment, the "horizontal width"[8] of that node in our tree equals (i.e., "marks") the *length of the expanded task list at the time the corresponding cycle of the busy period is executed*. A short reflection shows that this is true for any node in the tree.

Now, since it is obvious from the operation of FCFS scheduling that a busy period is T-feasible iff the length of the expanded task list is bounded by $T-1$ during the busy period, if follows that limiting the service times by a deadline T corresponds to limiting the "width" of the tree to $T-1$ vertices.

[8]The quotation stresses the fact that our width is not the usual width of a tree.

The ordinary generating function of this special family $\mathcal{B}_T$ of trees is simply the PGF $B_T(z)$ of the length of a feasible busy period. However, for the sake of computational simplicity it is preferable to deal with $\mathcal{C}_T = \mathcal{B}_{T+1}$. Similar to Section 4, we start with the symbolic equation



for all $T \geq 1$; $p_k = [z^k]P(z)$ have been defined in (2.3). The translation into generating functions reads

$$C_T(z) = \sum_{k=0}^{T} p_k z \prod_{j=T-k+1}^{T} C_j(z). \tag{5.1}$$

Defining

$$Q_n(z) = \frac{1}{C_n(z) \cdots C_1(z)} \qquad \text{and} \qquad Q_0(z) = 1,$$

we obtain $C_T(z) = Q_{T-1}(z)/Q_T(z)$ and hence

$$B_T(z) = \frac{Q_{T-2}(z)}{Q_{T-1}(z)}. \tag{5.2}$$

According to our preliminary discussions in Section 2, we need asymptotic results for $B_T(1)$ and $B'_T(1)$. Thus, we have to deal with the asymptotics of $Q_T(1)$ and $Q'_T(1)$. Note that quantities related to $Q_T(1)$ also arise in [**TA77**], remember our overview of related research in Section 2.

Multiplying our fundamental recurrence relation (5.1) by $Q_T(z)$ yields

$$Q_{T-1}(z) = z \sum_{k=0}^{T} p_k Q_{T-k}(z);$$

introducing the corresponding bivariate generating function, we find

$$Q(s,z) = \sum_{k \geq 0} Q_k(z) s^k = \frac{z p_0}{z P(s) - s}. \tag{5.3}$$

The investigation of $Q(s,z)$ reveals that the dominant singularity w.r.t. s is a polar one, resulting from zeroes of the function $P(s) - s$, that is, fixed points of $P(s)$.

In the normal case, it is not difficult to see that there is a trivial fixed point $s = 1$ and another one at $s = \kappa > 1$. Using Rouche's theorem, it may be shown that[9] there are no further fixed points within a disk of certain radius $R > \kappa$ around zero. Applying a straightforward singularity analysis based on subtracted singularities (simple poles at

[9] Actually, much more can be said, cf. Section 7, equation (7.4)ff.

$s = 1$ and $s = \kappa$) provides asymptotic expressions for $Q_T(1)$ and $Q'_T(1)$ for $T \to \infty$, eventually revealing

$$B_T(1) = 1 - \frac{(\kappa - 1)(1 - P'(1))}{P'(\kappa) - 1}\kappa^{-T} + O(R^{-T})$$
$$B'_T(1) = \frac{1}{1 - P'(1)} + O(T\kappa^{-T})$$

by (5.2). Remembering (3.3), some straightforward algebra finally provides

THEOREM 5.1. (FCFS scheduling in the normal case, cf. [**SB92**, Theorem 1]). *The successful T-run duration $\mathcal{S}_T$ for FCFS scheduling in the normal case is approximately exponentially distributed with parameter $1/\mu_T^{FCFS}$ where*

$$\mu_T^{FCFS} = \frac{P'(\kappa) - 1}{(\kappa - 1)(1 - P'(1))^2}\kappa^T(1 + O(1/T)) \qquad \text{for } T \to \infty.$$

$\kappa > 1$ is the solution of $x = P(x)$, $x > 1$. ∎

In the balanced case, it turns out that $\kappa = 1$. However, using a refined subtracted singularity analysis of $Q(s, z)$ (which now involves a higher-order pole at $s = 1$), appropriate asymptotic expressions for $B_T(1)$ and $B'_T(1)$ are readily obtained and we find

THEOREM 5.2. (FCFS scheduling in the balanced case, cf. [**BS91**, Theorem 3]). *The expectation of the successful T-run duration $\mathcal{S}_T$ for FCFS scheduling in the balanced case is given by*

$$\mu_T^{FCFS^b} \sim \frac{1}{\psi_i} \cdot \frac{i!}{(i-1)(2i-1)!}T^i \qquad \text{for } T \to \infty,$$

where $i \geq 2$ denotes the order of the zero of $P(x) - x$ at $x = 1$, i.e., the smallest integer value of i such that

$$P(x) - x = \psi_i(x - 1)^i + O\big((x - 1)^{i+1}\big) \qquad \text{for } x \to 1$$

and $\psi_i \neq 0$. ∎

Note that we also determined the variance in the balanced case, see [**BS91**] for details.

In the overloaded case, the fixed point $s = \kappa$ is less than the other one at $s = 1$, so that the dominant singularity is caused by $\kappa = \beta < 1$. A singularity analysis following the one for the normal case above eventually provides

THEOREM 5.3. (FCFS scheduling in the overloaded case, cf. [**BS91**, Theorem 1]). *The expectation of the successful T-run duration $\mathcal{S}_T$ for FCFS scheduling in the overloaded case is given by*

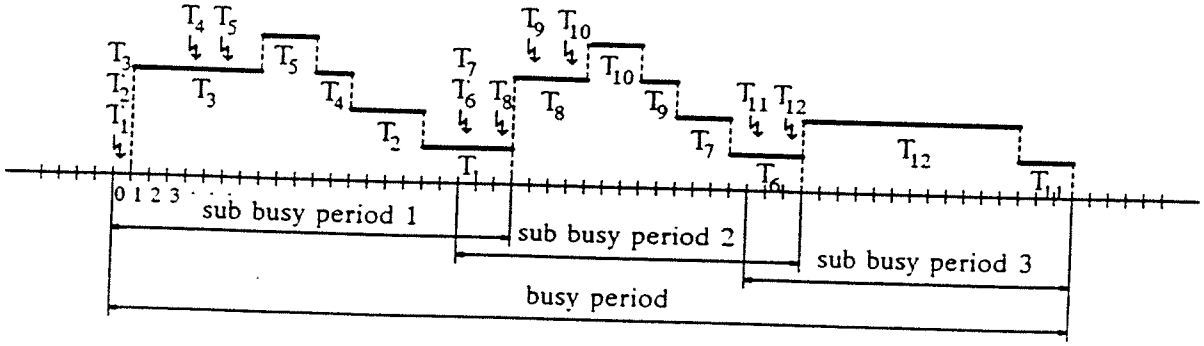
$$\mu_T^{FCFS^o} \sim \frac{\beta}{1 - \beta} \cdot \frac{1}{1 - P'(\beta)} \qquad \text{for } T \to \infty,$$

where $\beta < 1$ is the solution of $x = P(x)$, $x < 1$. ∎

6. Nonpreemptive LCFS Scheduling

This Section is devoted to the investigation of T-feasible busy periods in the case of nonpreemptive LCFS scheduling contained in [**SB94**]. The algorithm works as follows: If a task arrives at the system during the execution of another one, its execution is started when the latter has finished. If several tasks arrive during the execution of the same task, they are scheduled for execution one after the other, in an arbitrary —preferably LCFS— sequence. Hence, nonpreemptive LCFS may be viewed as preemptive LCFS where whole tasks (instead of single cycles) form the non-preemtible unit.

Consider the following example:

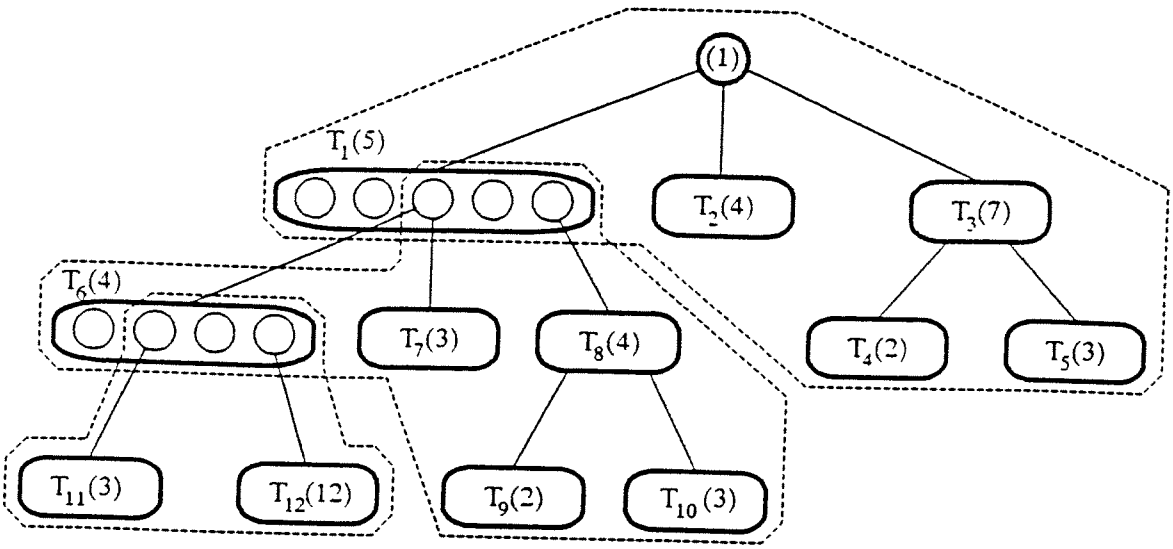


Again, the horizontal axis is divided into equidistant cycles. Those cycles forming the busy period of interest are numbered consecutively; cycle 0 denotes the initial (idle) cycle. Task arrivals are shown by small lightnings with tasknames above. The execution of a task is represented by a horizontal line whose length equals the task execution time. The vertical level of such a line, i.e., its vertical distance to the horizontal axis, represents the number of tasks not processed to completion at the beginning of the corresponding task, i.e., the number of preempted tasks plus 1. For readability, we attached the name of the appropriate task (and, if possible, its task execution time) to each line.

As in the case of preemptive LCFS scheduling, there is an important relation between deadline constraints and the length of *sub busy periods*. Here, a sub busy period denotes the epoch from the arrival of the first (new) task during the execution of a level 1 (or level 0) task to the end of the last cycle of that new task. (This definition is slightly different from the definition of a sub busy period in Section 4). For instance, looking at the cycle 0 in our example, one obtains the arrival of task T_1. Due to the nonpremptive LCFS scheduling discipline, this task is badly off, because all tasks arriving before the beginning of the execution of T_1 are prefered! Hence, if the length of a sub busy period is less or equal to T, all processed tasks are guaranteed to meet a service time deadline of T cycles. Conversely, if the length of a sub busy period is larger than T, at least the task having arrived first will miss its deadline.

Again, we will establish a suitable mapping between busy periods and a family of (labeled) random trees, which provides a straightforward correspondence between deadline constraints and limited label sums of some subtrees.

The following tree corresponds to our example above:



Each task is represented by an elliptical node which is labeled according to its task execution time, i.e., the label of a node is the number of cycles necessary for processing the task to completion. Equivalently, this labelling may be done by drawing the corresponding number of circles (◯, denoting a single action, of course) within the node. The number of successors of a node equals the number of arrivals during the execution of the corresponding task. Successors are drawn from the left to the right, according to their arrival sequence. Note that the reconstruction of the busy period from a given tree is done by a right-to-left preorder traversal of all (elliptical) nodes of the tree.

Due to our construction, the outer leftmost (elliptical) nodes in the tree correspond to those tasks which both complete a sub busy period and start a new one, too. They are displayed in the equivalent labeling-style mentioned above. If such a node has no successors, it indicates the end of the whole busy period; at least one idle cycle follows.

Deadline constraints are reflected by suitable limits on the number of cycles. More precisely, the sum of the labels of nodes belonging to a sub busy period has to be less than the deadline T, for all sub busy periods, of course. In our example above, those nodes belonging to a specific sub busy period are surrounded by a dotted line.

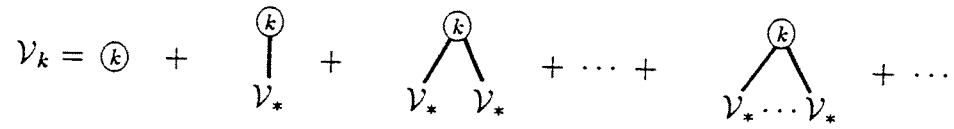
Unfortunately, the fact that consecutive sub busy periods overlap one another, introduces unpleasant difficulties. Since two consecutive sub busy periods are pasted together at an outer leftmost node, (some of) its cycles have to be taken into account in both. On the other hand, to obtain the total number of cycles of a whole busy period, each cycle has to be counted excatly once. Hence, we are forced to investigate trees representing sub busy periods first, and paste them together in order to obtain whole busy periods.

We start our treatment concerning $B_T(u)$ with the investigation of the family $\mathcal{B}_{i,j}$ of trees which correspond to sub busy periods starting with a label i node and completing with a label j node ($i \geq 1$, $j \geq 1$). To keep the symbolic equations simple, we defer attaching the necessary probability weights to the translation into generating functions.

We have the following decomposition:

$$\mathcal{B}_{i,j} = \mathcal{H}_j\mathcal{V}_{i-1} + \mathcal{E}\mathcal{H}_j\mathcal{V}_{i-2} + \mathcal{E}^2\mathcal{H}_j\mathcal{V}_{i-3} + \cdots + \mathcal{E}^{i-2}\mathcal{H}_j\mathcal{V}_1 + \mathcal{E}^{i-1}\mathcal{H}_j. \tag{6.1}$$

The combinatorial objects used for building blocks have straightforward meaning. $\mathcal{E}$ denotes a single cycle with no task arrivals, $\mathcal{H}_j$ denotes a single cycle with at least one arrival, leading to the leftmost label j node. $\mathcal{V}_k$ denotes a sequence of $k \geq 1$ consecutive cycles with an arbitrary number of arrivals. To start with the most important one, we have the following symbolic equation:

$\mathcal{V}_k =$  $+ \cdots$

with $\mathcal{V}_* = \sum_k \mathcal{V}_k$. Recalling definition (2.2), the ordinary generating function of $\mathcal{V}_*$ reads

$$V_*(z) = \sum_{k\geq 1} l_k V_k(z).$$

Due to definition (2.1), we have

$$q_{n,k} = \text{prob}\{n \text{ task arrivals during } k \text{ (consecutive) cycles}\} = [z^n]A(z)^k$$

for $n \geq 0$, $k \geq 1$. Thus, the OGF of $\mathcal{V}_k$ reads

$$V_k(z) = z^k \sum_{n\geq 0} q_{n,k} V_*(z)^n. \tag{6.2}$$

Introducing the bivariate generating function

$$G(z,u) = \sum_{k\geq 0} l_k V_k(z) u^k,$$

one obtains $V_*(z) = G(z,1)$. Multiplying (6.2) by $l_k u^k$ and summing up for $k \geq 1$ yields $G(z,u) = L\big(zuA(G(z,1))\big)$. Because of

$$V_1(z) = [l_1 u^1]G(z,u) = zA\big(G(z,1)\big), \tag{6.3}$$

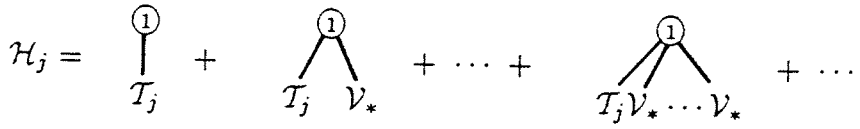
we find

$$\sum_{k\geq 1} l_k V_k(z) u^k = G(z,u) = L\big(uV_1(z)\big) = \sum_{k\geq 1} l_k V_1(z)^k u^k,$$

hence $V_k(z) = V_1(z)^k$ and $V_*(z) = G(z,1) = L\big(V_1(z)\big)$. Substituting the latter in (6.3) and introducing the abbreviation $B(z) = V_1(z)$, we obtain a result already known from Section 4:

$$B(z) = zP\big(B(z)\big).$$

At next, we look at $\mathcal{H}_j$, $j \geq 1$. The symbolic equation reads



with $\mathcal{T}_j$ denoting a label j node. Obviously, the corresponding OGF is $T_j(z) = l_j z^j$.

Since each combinatorial object in $\mathcal{H}_j$ corresponds to an object in $\mathcal{V}_1$, where the leftmost successor $\mathcal{V}_*$ (at the top level) is replaced by $\mathcal{T}_j$, we may omit the detailed translation of the symbolic equation and write down the result immediately:

$$H_j(z) = l_j z^j \frac{V_1(z) - a_0 z}{L\big(V_1(z)\big)}.$$

Note that the term $a_0 z$ corresponds to the 'smallest' tree in $\mathcal{V}_1$, which consists of the root only (no arrivals during the corresponding cycle).

The OGF for $\mathcal{E}$ is straightforward; mentioning definition (2.1), we have $E(z) = a_0 z$.

Now we are able to translate symbolic equation (6.1) into the appropriate OGF. For reasons that will become evident when pasting sub busy periods together, we attach two different sizes to a structure of that class. Roughly speaking, the size represented by z is responsible for counting the length of the corresponding sub busy period w.r.t. deadline properties. A different size is represented by the variable u. It counts the contributions of the corresponding sub busy period to the overall length of the whole busy period; remember our remarks at the end of Section 2. We find

$$B_{i,j}(z,u) = \sum_{l=0}^{i-1} E(1)^{i-1-l} H_j(zu) B(zu)^l u^{-1-l}. \tag{6.4}$$

Note that we should have no contributions from $\mathcal{E}$, both for deadline counting and the overall size, thus $E(1)$ is used. The last term u^{-1-l} makes the difference in the size counted by z and u. The $l+1$ cycles within the initial label i node, i.e., the 'roots' of $\mathcal{H}_j$ and $\mathcal{V}_1^l$, must be counted in z only (deadlines), not in u. The latter is done in the preceding sub busy period!

That is, for a sub busy period starting with a label i and terminating with a label j node, $[z^t][u^n]B_{i,j}(z,u)$ is the probability that all tasks meet a deadline of t cycles (and no smaller one), contributing n cycles to the length of the whole busy period.

Evaluating expression (6.4) yields

$$B_{i,j}(z,u) = \left(\left(\frac{B(zu)}{u} \right)^i - a_0^i \right) \frac{B(zu) - a_0 zu}{L(B(zu))} \cdot \frac{1}{B(zu) - u a_0} l_j (zu)^j. \tag{6.5}$$

Now, we will try to paste sub busy periods together. In order to enable deadline counting in each sub busy period, we are forced to use *different counting variables* z_k instead of z. Let $\mathcal{B}_{i,j}^k$ denote the family of trees, which are formed by pasting together exactly $k \geq 1$ sub busy periods. For example, we have

$$\mathcal{B}_{i,j}^2 = \sum_{k \geq 1} \mathcal{B}_{i,k} \mathcal{B}_{k,j},$$

the corresponding (multivariate) generating function reads

$$B^2_{i,j}(z_2, z_1; u) = \sum_{k \geq 1} B_{i,k}(z_2, u) B_{k,j}(z_1, u).$$

For simplicity, we introduce the abbreviations $B_{i,j}(z, u) = S_i(z, u) I(z, u) T_j(z, u)$, cf. equation (6.5), and obtain

$$B^2_{i,j}(z_2, z_1; u) = S_i(z_2, u) I(z_2, u) \big[L(z_2 B(z_1 u)) - L(a_0 z_2 u) \big] I(z_1, u) T_j(z_1, u).$$

Note that overlapping of sub busy periods is reflected by the 'connecting function' within the brackets. The 'starting' and 'trailing' functions $S_i(., u)$ and $T_j(., u)$ appear in the expression again; thus we may use this technique repeatedly to construct the general term:

$$\begin{aligned} B^k_{i,j}(z_k, \ldots, z_1; u) = S_i(z_k, u) I(z_k, u) \cdot & \\ & \big(L(z_k B(z_{k-1} u)) - L(a_0 z_k u) \big) I(z_{k-1}, u) \cdot \\ & \big(L(z_{k-1} B(z_{k-2} u)) - L(a_0 z_{k-1} u) \big) I(z_{k-2}, u) \cdot \\ & \vdots \\ & \big(L(z_2 B(z_1 u)) - L(a_0 z_2 u) \big) I(z_1, u) \cdot \\ & T_j(z_1, u). \end{aligned}$$

To construct a whole busy period consisting of exactly k sub busy periods, we have to deal with the decomposition

$$C^k = \mathcal{U} \sum_{j \geq 1} \mathcal{B}^k_{1,j} \mathcal{E}_j.$$

$\mathcal{U}$ denotes a single cycle forming the initial cycle of the first sub busy period, its OGF is $U(z) = z$. $\mathcal{E}_j$ is a label j node with no arrivals; we have the OGF $E_j(z) = E(z)^j = (a_0 z)^j$. Translating the symbolic equation above, we find

$$C^k(z_k, \ldots, z_1; u) = u \sum_{j \geq 1} B^k_{1,j}(z_k, \ldots, z_1; u) E(1)^j.$$

Note that we do not count cycles resulting from the terminating idle period, i.e., $\mathcal{E}_j$. We easily obtain

$$\begin{aligned} C^k(z_k, \ldots, z_1; u) = u \Big(\frac{B(z_k u)}{u} - a_0 \Big) & \frac{B(z_k u) - a_0 z_k u}{L\big(B(z_k u)\big)} \cdot \frac{1}{B(z_k u) - u a_0} \cdot \\ & \big(L(z_k B(z_{k-1} u)) - L(a_0 z_k u) \big) \frac{B(z_{k-1} u) - a_0 z_{k-1} u}{L\big(B(z_{k-1} u)\big)} \cdot \frac{1}{B(z_{k-1} u) - u a_0} \cdot \\ & \vdots \\ & \big(L(z_2 B(z_1 u)) - L(a_0 z_2 u) \big) \frac{B(z_1 u) - a_0 z_1 u}{L\big(B(z_1 u)\big)} \cdot \frac{1}{B(z_1 u) - u a_0} \cdot \\ & L(a_0 z_1 u). \end{aligned}$$

Obviously, a busy period with no sub busy periods, that is, an idle cycle, has the symbolic equation $\mathcal{U}\mathcal{E}_1$. The corresponding OGF is very simple: $C^0(u) = a_0 u$.

Since a whole busy period may consist of an arbitrary number of sub busy periods not exceeding T cycles (for deadline counting, of course), we are forced to study

$$B_T(u) = a_0 u + \sum_{k\geq 1} [z_k^T] \cdots [z_1^T] \frac{1}{1-z_k} \cdot \frac{1}{1-z_{k-1}} \cdots \frac{1}{1-z_1} C^k(z_k, \ldots, z_1; u),$$

which is the PGF of the length of an arbitrary busy period containing no deadline violation. Introducing the abbreviations $y_k = z_k u$ and

$$\begin{aligned} D^k(y_k, \ldots, y_1; u) &= \frac{1}{1-y_k/u} \cdot \frac{1}{1-y_{k-1}/u} \cdots \frac{1}{1-y_1/u} C^k(y_k/u, \ldots, y_1/u; u) \\ &= \frac{1}{1-y_k/u} \cdot \frac{B(y_k) - a_0 y_k}{L(B(y_k))} \cdot \\ &\quad \frac{1}{1-y_{k-1}/u} \cdot \frac{L(\frac{y_k}{u} B(y_{k-1})) - L(a_0 y_k)}{B(y_{k-1}) - u a_0} \cdot \frac{B(y_{k-1}) - a_0 y_{k-1}}{L(B(y_{k-1}))} \cdot \\ &\quad \vdots \\ &\quad \frac{1}{1-y_1/u} \cdot \frac{L(\frac{y_2}{u} B(y_1)) - L(a_0 y_2)}{B(y_1) - u a_0} \cdot \frac{B(y_1) - a_0 y_1}{L(B(y_1))} \cdot \\ &\quad L(a_0 y_1) \end{aligned}$$

we find

$$B_T(u) = a_0 u + \sum_{k\geq 1} u^{kT} [y_k^T] \cdots [y_1^T] D^k(y_k, \ldots, y_1; u). \tag{6.6}$$

Looking more closely at the delicate expression for $B_T(u)$ (where u is assumed to be a complex parameter in the closed disk $\mathcal{D}(1,\nu) = \{z : |z-1| \leq \nu\}$ for some arbitrary small $\nu > 0$), one obtains non-trivial interdependencies among the coefficients $[y_k^T], \ldots, [y_1^T]$, resulting from the "connecting functions" $L(\frac{y_i}{u} B(y_{i-1}))$. Hence, a direct extraction of the desired coefficients yields complicated expressions, at first (and possibly second) sight far away from tractability. Thus, we will use a somewhat elaborate singularity analysis technique which we called *asymptotic separation*: Although it is impossible to separate the "connecting functions" directly, i.e., to split up $L(\frac{y_i}{u} B(y_{i-1}))$ into a product $f(y_i)g(y_{i-1})$, it is possible to provide a separable asymptotic expansion which serves as well. However, this needs some explanation:

Looking more closely at y_1-related terms in $B_T(u)$, that is

$$\frac{1}{1-y_1/u} \cdot \frac{L(\frac{y_2}{u} B(y_1)) - L(a_0 y_2)}{B(y_1) - u a_0} \cdot \frac{B(y_1) - a_0 y_1}{L(B(y_1))} \cdot L(a_0 y_1), \tag{6.7}$$

our task is the determination of the T-th Taylor coefficient $[y_1^T]$ in this multivariate function, which is analytic for y_1, y_2 in a neighborhood of 0 and $u \in \mathcal{D}(1,\nu)$. Due to general

theorems (Cauchy's formula for multivariate analytic functions, cf. [**Mar65**, p. 101ff]), $[y_1^T]f(y_2, y_1, u)$ is an analytic function of y_2 and u, too. In addition, it is not hard to prove that the well-known *transfer lemmas* (see [**FO90**]) remain valid for multivariate analytic functions. For example, if

$$f(z,w) = O\big(g(w)(1-z/\zeta)^{\alpha}\big) \qquad \text{for } z \to \rho,$$

uniformly w.r.t. w, it follows that

$$[z^n]f(z,w) = O\big(g(w)n^{-1-\alpha}\zeta^{-n}\big) \qquad \text{for } n \to \infty,$$

uniformly in w, too. Again, keep in mind that the latter $O(.)$ represents a function which is analytic in w!

Returning to our original function, we obtain three "sources" of singularities,

(1) a (removeable) simple pole at $y_1 = \zeta(u) < 1$, resulting from $B\big(\zeta(u)\big) = a_0 u$,
(2) a simple pole at $y_1 = u$,
(3) an algebraic singularity at $y_1 = \rho$ resulting from functions involving $B(y_1)$, cf. the comments on expansion (4.4).

The fact that $y_1 = \zeta(u)$ is a removeable singularity, i.e., that there is no singularity at all, is easily established by taking into account the zero of $L(\frac{y_2}{u}B(y_1)) - L(a_0 y_2)$ at $y_1 = \zeta(u)$.

Remembering $\rho > 1$ it follows that $y_1 = u$ is the singularity with the smallest modulus; in fact we only have to choose ν small enough, i.e., $1+\nu < \rho$. The appropriate contribution to $[y_1^T]$ is easily determined via *subtracted singularities*:

$$\frac{L(\frac{y_2}{u}B(u)) - L(a_0 y_2)}{L\big(B(u)\big)} L(a_0 u) \cdot u^{-T}.$$

Investigating the behaviour of (6.7) near the "next" singularity $y_1 = \rho$ it turns out that $B(y_1) - a_0 y_1$ and $L\big(B(y_1)\big)$ obey expansions similar to $B(y_1)$. The function $L(a_0 y_1)$ is assumed to have a radius of convergence larger than ρ, i.e., should be well-behaved in a neighborhood of $y_1 = \rho$. Hence, the only remaining difficulty concerns the term containing the "connecting function", i.e.,

$$\frac{L(\frac{y_2}{u}B(y_1)) - L(a_0 y_2)}{B(y_1) - u a_0}.$$

But, using the mentioned extension of a transfer lemma it is possible to attack this multivariate analytic function as well. Since y_1 comes up with $B(y_1)$, one feels that $L(\frac{y_2}{u}B(y_1))$ should have an algebraic singularity at $y_1 = \rho$, independent of y_2! Due to the fact that, at our next 'stage', y_2 will play the role of y_1 and y_3 the one of y_2, it is obvious to ask for the behaviour in a neighborhood of $y_2 = \rho$ (and also $y_2 = u$ to account for the subtracted singularity term for y_2). However, since y_2 appears in conjunction with the well-behaved function $L(.)$ only, we may expect inferior influences here. To make a long story short, we assert that it is possible to determine a uniform expansion

$$\frac{L(\frac{y_2}{u}B(y_1)) - L(a_0 y_2)}{B(y_1) - u a_0} = b(y_2,u) + c(y_2,u)(1-y_1/\rho)^{1/2} + O(1-y_1/\rho) \qquad \text{for } y_1 \to \rho$$

where $b(y_2,u)$ and $c(y_2,u)$ denote well-behaved analytic functions of both y_2 and u. The remainder $O(1-y_1/\rho)$ represents a multivariate analytic function, too, and the implied constant is independent of y_1, y_2 and u.

Putting all terms together, we obtain a uniform expansion for (6.7) at $y_1=\rho$, similar to the expansion above:

$$\beta(y_2,u)L(a_0\rho)+\gamma(y_2,u)L(a_0\rho)(1-y_1/\rho)^{1/2}+O(1-y_1/\rho) \quad \text{for } y_1\to\rho.$$

Note that the terms $L(a_0\rho)$ represent the contribution resulting from the terminating function $L(a_0y_1)$.

The subtracted term resulting from the simple pole $y_1=u$ is meaningless for the analysis of the singularity $y_1=\rho$ since $(1-y_1/u)^{-1}$ is analytic for all $y_1\neq u$. Using transfer lemmas, the desired coefficient $[y_1^T]$ finally yields

$$\alpha(y_2,u)L(a_0u)u^{-T}-\frac{1}{2\sqrt{\pi}}\gamma(y_2,u)L(a_0\rho)T^{-3/2}\rho^{-T}+O(T^{-2}\rho^{-T}) \qquad \text{for } T\to\infty$$

with both $\alpha(y_2,u)$ and $\gamma(y_2,u)$ analytic at $y_2=\rho$; the 'elimination' of y_1 is complete.

Now, the same procedure may be used for the extraction of $[y_2^T]$ (hence, for all $[y_i^T]$) since the related terms are almost the same as before. In fact, the only difference springs from replacing $L(a_0y_1)$ by $\alpha(y_2,u)$ and $\gamma(y_2,u)$, respectively! Using this simple iterative scheme (leading to a recurrence relation) it is possible to compute an asymptotic expansion

$$B_T(u)=B(u)-R(u)u^TT^{-3/2}\rho^{-T}+O(u^TT^{-2}\rho^{-T})$$

uniformly valid for $u\in\mathcal{D}(1,\nu)$. By virtue of a general theorem concerning uniform expansions we may differentiate this expansion in order to derive $B_T^{(m)}(1)$ for an arbitrary but fixed m.

We therefore obtain

THEOREM 6.1. (nonpreemptive LCFS scheduling in the normal case, cf. [**SB94**, Theorem 5.1]). *The successful T-run duration S_T for nonpreemptive LCFS scheduling in the normal case is approximately exponentially distributed with parameter $1/\mu_T^{npLCFS}$ where*

$$\mu_T^{npLCFS}=C\,T^{3/2}\rho^T\big(1+O(1/T)\big) \qquad \text{for } T\to\infty,$$

and

$$C=\frac{2\sqrt{\pi}(\rho-1)(\tau-a_0)L(\tau)}{bL(\rho)(1-P'(1))}\left(\frac{(1-a_0)\,(L(\tau)-L(a_0))\,a_0(\rho-1)}{L(a_0)(\tau-a_0)}-\frac{(\tau-1)(\tau-a_0\rho)L'(\tau)}{L(\tau)}+\tau-a_0\right)^{-1},$$

$\tau>1$ is the solution of $P(x)=xP'(x)$, $\rho=\tau/P(\tau)>1$, and $b=\sqrt{2P(\tau)/P''(\tau)}$. ∎

Unfortunately, our complicated asymptotic analysis above is not valid for the balanced case since $\rho=1$. However, using a refined analysis it is possible to show

THEOREM 6.2. (nonpreemptive LCFS scheduling in the balanced case). *The expectation of the successful T-run duration S_T for nonpreemptive LCFS scheduling in the balanced case is given by*

$$\mu_T^{npLCFS^b} \sim T \qquad \text{for } T \to \infty. \quad \blacksquare$$

In the overloaded case, the result of the asymptotic analysis concerning the normal case is valid again since $\rho > 1$ (though $\tau < 1$). Due to the fact $B(1) < 1$ we eventually obtain

THEOREM 6.3. (nonpreemptive LCFS scheduling in the overloaded case). *The expectation of the successful T-run duration S_T for nonpreemptive LCFS scheduling in the overloaded case is given by*

$$\mu_T^{npLCFS^o} \sim \frac{\beta}{1-\beta} \cdot \frac{1}{1-P'(\beta)} \qquad \text{for } T \to \infty,$$

where $\beta < 1$ is the solution of $x = P(x)$, $x < 1$. $\blacksquare$

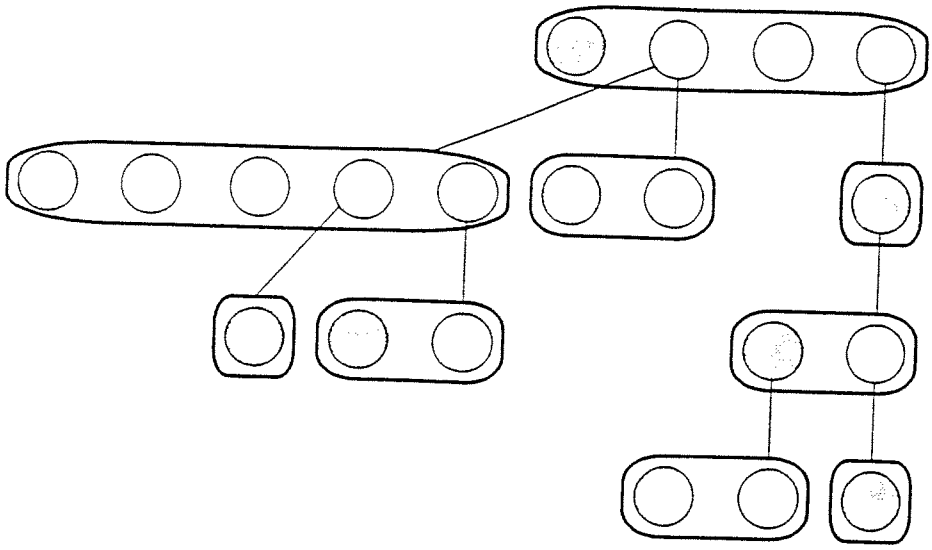
7. STATIC PRIORITY SCHEDULING

This section is devoted to a (very brief) survey of the rather involved analysis of the widespreadly used static priority scheduling algorithm (SPS) contained in [**S94**]. Unlike the no-priority algorithms described so far, we have now $L \geq 1$ different classes of tasks with different priority levels, numbered from $1, \ldots, L$, where 1 is the highest one. For each priority level ℓ, there is an associated (constant) deadline T_ℓ and two probability generating functions $A_\ell(z)$ (arrivals) and $L_\ell(z)$ (task execution times); again, we introduce

$$P_\ell(z) = \sum_{k \geq 0} p_{k,\ell} z^k = A_\ell\big(L_\ell(z)\big). \tag{7.1}$$

Static priority scheduling works as follows: Assuming that the task list of our system is sorted according to descending priorities, a newly arriving task of a certain priority level is inserted into the queue behind the already queued tasks of the same level. The server always executes the task at the head of the task list, in a preemptible fashion. Note however that any scheduling takes place at cycle boundaries only, i.e., no preemption occurs during a cycle.

Our combinatorial analysis relies on an extension of the ideas used for FCFS scheduling (which of course corresponds to SPS with $L = 1$): Since tasks of the same priority level are queued in FCFS order, whereas higher priority ones are handeled according to the (preemptive) LCFS policy, we just have to "blow" up nodes of the trees from Section 5 in such a way that each of the resulting *multi-nodes* corresponds to a single higher priority busy period. This naturally implies that the width of such a multi-node is greater or equal to 1; moreover, we have to take into account the appropriate probabilities in the probability weights of the vertices. The following tree shows an example for $L = 2$:



The (higher-level) actions corresponding to the vertices within a multi-node are executed from left to right. Note however that there is exactly one (shaded) node within each multi-node, which belongs to a level-2 task; it represents the initial cycle of the corresponding level-1 busy period. Since such cycles are obviously executed *before* their higher-priority neighbors, it is evident that all outer leftmost multi-nodes in our tree require special attention: They may exceed the (lower-priority) width-constraint provided that there are no lower-priority arrivals in those nodes of the multi-node which lie beyond the limit.

First we consider the (of course well-known) case where no deadline restrictions are present. Here, we have the following symbolic equation for the family $\mathcal{B}^{(L)}$ of corresponding trees (note that those trees do not reflect the execution order of actions!):

$$\mathcal{B}^{(L)} = p_0^{(L)} \bigcirc \; + \; p_1^{(L)} \; \underset{\mathcal{B}^{(L)}}{\overset{\bigcirc}{|}} \; + \cdots \; p_k^{(L)} \; \underbrace{\overset{\bigcirc}{\diagup\diagdown}\; \mathcal{B}^{(L)} \ldots \mathcal{B}^{(L)}}_{k} \; + \cdots .$$

The probability weights $p_k^{(L)}$ denote the probability that the total number of actions to be executed due to task arrivals during the initial cycle (denoted by $\bigcirc$) equals k. Since arrivals and task execution times (of different priority levels) are independent, we obviously have

$$p_k^{(L)} = [z^k]P^{(L)}(z) = [z^k]\prod_{\ell=1}^{L} P_\ell(z) \qquad \text{for all } k \geq 0.$$

Applying straightforward product and sum translations, cf. [**VF90**], it is easy to see that the PGF $B^{(L)}(z)$ of arbitrary busy periods solves the already well-known functional equa-

tion (cf. equation (4.3), [MM78])

$$B^{(L)}(z) = z \sum_{k \geq 0} p_k^{(L)} \big(B^{(L)}(z)\big)^k = z P^{(L)}\big(B^{(L)}(z)\big).$$

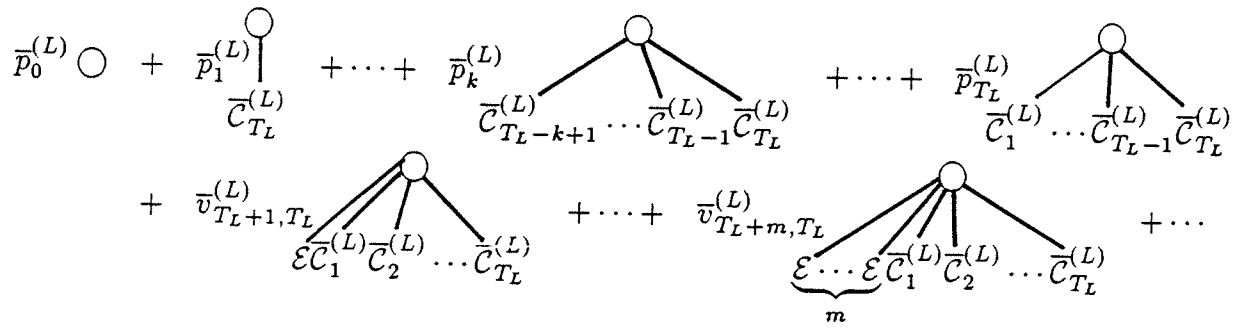
Now we will proceed with the derivation of the OGFs of interest. For technical reasons, it is more convenient to deal with the family of trees which are width-constrained by T_ℓ instead of $T_\ell - 1$, shifting back everything by 1 at the end. Thus, we consider the family $C^{(L)}_{T_L,\ldots,T_1}$ of $T_1, \ldots, T_L$-width trees with the OGF (which are improper PGFs in this case)

$$C^{(L)}_{T_L}(z) = C^{(L)}_{T_L,\ldots,T_1}(z) = \sum_{n \geq 0} c^{(L)}_{n,T_L} z^n = \sum_{n \geq 0} c^{(L)}_{n,T_L,\cdots,T_1} z^n.$$

Note that obviously $c^{(L)}_{0,T_L} = 0$, because there is always a root node in any tree. For notational convenience, we use abbreviations like the above ones where possible.

Our aim is to provide an equation for $C^{(L)}_{T_L,\ldots,T_1}(z)$ which involves $C^{(L-1)}_{T_{L-1},\ldots,T_1}(z)$, i.e., a recursive formula. However, we first derive a symbolic equation of similar width-constrained trees $\overline{C}^{(L)}_{T_L} = \overline{C}^{(L)}_{T_L,\ldots,T_1}$ with L priority levels, which are generated by level-L arrivals during a *single* initial cycle; the connection to the actually desired family $C^{(L)}_{T_L} = C^{(L)}_{T_L,\ldots,T_1}$ will be established subsequently.

$\overline{C}^{(L)}_{T_L} =$



In the equation above, $\mathcal{E}$ denotes a single cycle with no level-L arrivals; its OGF is clearly $E(z) = p_{0,L} z$, cf. (7.1).

It is easy to provide the required probability weights $\overline{p}^{(L)}_k$, which denote the probability that exactly $k \leq T_L$ new actions arise as a consequence of (1) $T_L \geq m \geq 0$ level-L arrivals during the initial cycle and (2) all higher-priority arrivals during the m arising (level-L-)successors of the initial cycle. With $C^{(L-1)}_{T_{L-1}}(z) = C^{(L-1)}_{T_{L-1},\ldots,T_1}(z)$ denoting the OGF of width-constrained trees for higher priority levels $L-1, \ldots, 1$, it is clear that

$$\overline{p}^{(L)}_k = [z^k] P_L\big(C^{(L-1)}_{T_{L-1}}(z)\big) \qquad \text{for } k \geq 0;$$
$$\overline{p}^{(L)}_0 = p_{0,L}$$

However, it is more complicated to provide the probability weights $\overline{v}^{(L)}_{k,T_L}$, $k > T_L \geq 1$, which reflect the situation of outer leftmost multi-nodes exceeding a width of T_L, as

mentioned earlier. For our argument, we use two counting variables y, w to take care of all cycles (y) and those which are meaningful for level-L deadlines only (w). Now, while all inner multi-nodes that arrived during the initial cycle contribute to both y and w, we have to replace the outer leftmost multi-node $C_{T_{L-1}}^{(L-1)}(yw)$ by a special multi-node, say, $C(w,y)$, that contributes different to w and y. More specifically, we require

$$C(w,y) = \sum_{n\geq 1} w^n \sum_{k\geq n} c_{n,T_{L-1}}^{(L-1)} y^k = \frac{w\big(C_{T_{L-1}}^{(L-1)}(y) - C_{T_{L-1}}^{(L-1)}(yw)\big)}{1-w}$$

so that $[y^k][w^n]C(w,y) = c_{k,T_{L-1}}^{(L-1)}$ if $k \geq n \geq 1$ (and zero otherwise). For $k > n \geq 1$, it is not difficult to find

$$\begin{aligned}\overline{v}_{k,n}^{(L)} &= \text{prob}\{k \text{ actions arise with } n \text{ being the last level-}L \text{ ``admissible'' one}\} \\ &= [y^k][w^n]\frac{P_L\big(C_{T_{L-1}}^{(L-1)}(yw)\big) - p_{0,L}}{C_{T_{L-1}}^{(L-1)}(yw)} \cdot \frac{w}{1-w} \cdot C_{T_{L-1}}^{(L-1)}(y).\end{aligned}$$

Note that we may discard the term $C_{T_{L-1}}^{(L-1)}(yw)$ since $[y^l][w^m]\frac{w}{1-w}C_{T_{L-1}}^{(L-1)}(yw) = 0$ for $l \geq m$.

The translation of the symbolic equation for $\overline{C}_{T_L}^{(L)}$ into a functional equation of the OGFs involved yields

$$\overline{C}_{T_L}^{(L)}(z) = z\sum_{k=0}^{T_L} \overline{p}_k^{(L)} \prod_{i=T_L-k+1}^{T_L} \overline{C}_i^{(L)}(z) + z\prod_{i=1}^{T_L} \overline{C}_i^{(L)}(z) \sum_{m\geq 1} \overline{v}_{T_L+m,T_k}^{(L)} (zp_{0,L})^m.$$

Defining

$$\overline{Q}_n^{(L)}(z) = \frac{1}{\overline{C}_n^{(L)}(z)\cdots\overline{C}_1^{(L)}(z)} \qquad \text{and} \qquad \overline{Q}_0^{(L)}(z) \equiv 1,$$

remember Section 5, and the corresponding bivariate generating function

$$\overline{Q}^{(L)}(s,z) = \sum_{k\geq 0} \overline{Q}_k^{(L)}(z) s^k,$$

we multiply the above equation by $\overline{Q}_{T_L}^{(L)}(z)$ to obtain

$$\overline{Q}_{T_L-1}^{(L)}(z) = z\sum_{k=0}^{T_L} \overline{p}_k^{(L)} \overline{Q}_{T_L-k}^{(L)}(z) + z\sum_{m\geq 1} \overline{v}_{T_L+m,T_L}^{(L)} (zp_{0,L})^m.$$

This primarily involves a simple Cauchy product; after some (rather involved) algebra, we obtain

$$\overline{Q}^{(L)}(s,z) = \frac{Q^{(L)}(s,z) - H^{(L)}(s,z)}{C_{T_{L-1}}^{(L-1)}(s)},$$

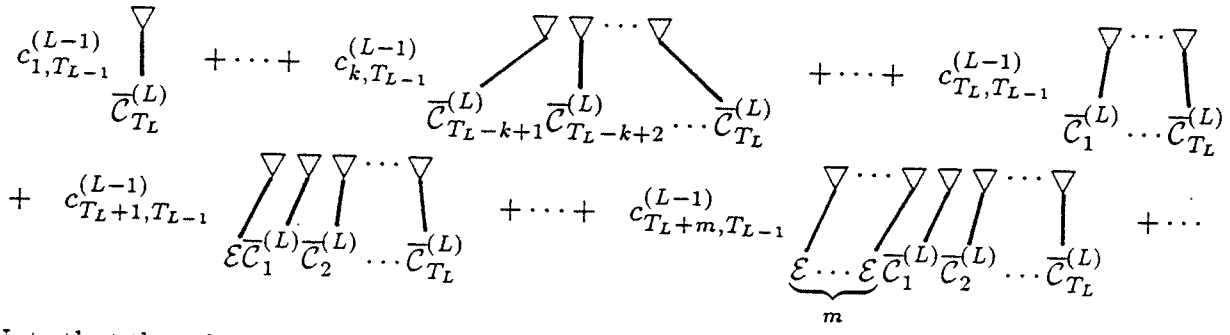
where

$$Q^{(L)}(s,z) = \sum_{k\geq 0} Q_k^{(L)}(z)s^k = \frac{sC_{T_{L-1}}^{(L-1)}(zp_{0,L})}{zP_L\left(C_{T_{L-1}}^{(L-1)}(s)\right) - s} \tag{7.2}$$

$$H^{(L)}(s,z) = \frac{sC_{T_{L-1}}^{(L-1)}(zp_{0,L}) - zp_{0,L}C_{T_{L-1}}^{(L-1)}(s)}{zp_{0,L} - s}. \tag{7.3}$$

In order to obtain the actually desired width-constrained trees $C_{T_L}^{(L)}$, we must take into account the possibility of higher-priority arrivals in the initial cycle (cf. our example tree at the beginning). This is accomplished by

$C_{T_L}^{(L)} =$



Note that the triangular nodes above are not to be counted in the OGF, since they represent ("double") the initial cycle of the associated successor trees only.

The translation into a functional equation of the OGFs involved yields

$$C_{T_L}^{(L)}(z) = \sum_{k=0}^{T_L} c_{k,T_{L-1}}^{(L-1)} \prod_{i=T_L-k+1}^{T_L} \overline{C}_i^{(L)}(z) + \prod_{i=1}^{T_L} \overline{C}_i^{(L)}(z) \sum_{m\geq 1} c_{T_L+m,T_{L-1}}^{(L-1)} (zp_{0,L})^m.$$

which may be attacked in the same way as the former one. Some algebra finally yields

$$C_{T_L}^{(L)}(z) = \frac{[s^{T_L}]Q^{(L)}(s,z)}{[s^{T_L}]\overline{Q}^{(L)}(s,z)},$$

and shifting back everything to arrive at the improper PGF of $T_1,\ldots,T_L$-feasible busy again, i.e., $B_{T_L,\ldots,T_1}^{(L)}(z) = C_{T_L-1,\cdots,T_1-1}^{(L)}(z)$, our major result follows:

THEOREM 7.1. (cf. [S94, Theorem 3.1]) *The improper PGF of $T_1,\ldots,T_L$-feasible busy periods may be expressed recursively by*

$$B_{T_L}^{(L)}(z) = \frac{[s^{T_L-1}]Q^{(L)}(s,z)}{[s^{T_L-1}]Q^{(L)}(s,z)/B_{T_{L-1}}^{(L-1)}(s) - [s^{T_L-1}]H^{(L)}(s,z)/B_{T_{L-1}}^{(L-1)}(s)} \qquad \text{for } L \geq 1,$$

$$B^{(0)}(z) = z,$$

with $Q^{(L)}(s,z)$ *and* $H^{(L)}(s,z)$ *defined by (7.2) and (7.3) (however, with* $B^{(L-1)}_{T_{L-1}}$ *replacing* $C^{(L-1)}_{T_{L-1}}$*).* ∎

This result covers the single priority level case ($L = 1$), that is, FCFS scheduling from Section 5 as well. Note that it is possible to show that $B^{(L)}_{T_L}(z)$ for $T_1, \cdots, T_L$ all being finite is a rational function.

The expression established in Theorem 7.1 may be attacked by asymptotic methods relying on (bivariate) singularity analysis, which eventually provide a (differentiable) uniform asymptotic expansion for $B^{(L)}_{T_L}(z)$ valid for $z \in \mathcal{D}(1,\varepsilon)$, ε sufficiently small. However, the appropriate analysis follows a different line as the one for FCFS scheduling, cf. Section 5, and we will only give the general line of reasoning due to lacking space. Note however, that the detailled asymptotic analysis is quite delicate and involves some interesting (and puzzling) mathematical problems.

It is easy to see, cf. (7.2), that the dominant singularities in the expression of Theorem 7.1 are determined by the solutions the already well-known functional equation (4.3), namely

$$F(s,z) = zU(s) - s = 0, \tag{7.4}$$

where $U(s)$ denotes an analytic function with certain properties, e.g., that the radius of convergence R_U of $U(z)$ must fulfill $R_U > 1$. This type of functional equation has been studied extensively (and controversially) in the literature, cf. [**MM78**], [**Can84**], [**MM89**] for some references. In general, it has several analytic solutions, but one is usually only interested in a particular one (with positive Taylor coefficients). We, however, require some results on other solutions as well. Therefore, we need novel proof techniques based on complex analysis, since "traditional" ones (e.g., [**MM89**]) rely heavily on the *a priori* assumption of positive coefficients.

From the properties of $U(s)$ it follows that

$$xU'(x) - U(x) = 0 \tag{7.5}$$

has a minimal positive solution $0 < \tau < R_U$, i.e., $\tau U'(\tau) - \tau = 0$. Defining

$$\rho = \frac{\tau}{U(\tau)}, \tag{7.6}$$

it is easily checked that $F(\tau,\rho) = 0$, $F_s(\tau,\rho) = 0$, and $F_{ss}(\tau,\rho) > 0$, where $F_s(s,z)$ and $F_{ss}(s,z)$ denotes the first and second partial derivative w.r.t. s, respectively. This in fact gives raise to the following well-known lemma, remember (4.4) in Section 4:

LEMMA 7.2. (cf. [**S94**, Lemma 4.1]) *With the properties and notations (7.4)–(7.6), the functional equation* $F(s,z) = zU(s) - s = 0$ *has a double-valued solution* $s = \chi(z)$ *for* z *in a neighborhood of* $z = \rho$ *(which is not necessarily the only one) and*

$$\chi(z) = \tau - \beta \cdot (1 - z/\rho)^{1/2} + \gamma \cdot (1 - z/\rho) + O\big((1 - z/\rho)^{3/2}\big) \qquad \text{for } z \to \rho;$$

β and γ are explicitely expressible in terms of derivatives of $U(s)$. ∎

Since $\chi(z)$ is double-valued near ρ, it is possible to define two branches $\zeta(z), \kappa(z)$, which are single-valued and analytic in a suitable small neighborhood of any $z_0 \neq \rho$. Thus, $F(s,z) = 0$ has two single-valued, analytic solutions in such a neighborhood. However, it is important to note that this result does not imply that there are no other solutions of $F(s,z) = 0$, mapping the neighborhood of $z = z_0$ to a different neighborhood $s = s_0'$, cf. [**Can84**] for an example. Nevertheless, the following lemma establishes that there are no further solutions, even in the case of arbitrary $z_0 = \alpha$, $0 < \alpha < \rho$, if we restrict ourselves to a certain domain of s.

LEMMA 7.3. (cf. [**S94**, Lemma 4.2]) *Let $U(z)$ be in accordance with (7.4)–(7.6). Then for any α, $0 < \alpha < \rho$ arbitrary but fixed there is some r_α, $\tau < r_\alpha < R_U$ such that $F(s,z) = zU(s) - s = 0$ restricted to the closed disk $s \in \overline{\mathcal{D}}(0, r_\alpha)$ has exactly two single-valued, analytic solutions $s = \zeta(z)$ and $s = \kappa(z)$, with values lying entirely in the interior of $\overline{\mathcal{D}}(0, r_\alpha)$ for every $z \in \mathcal{D}(\alpha, \varepsilon)$, $\varepsilon > 0$ sufficiently small. Moreover, $\zeta(x)$ and $\kappa(x)$ are positive real-valued for real positive $0 < x < \rho$, satisfying $\zeta(x) < \tau$ and $\kappa(x) > \tau$.* ∎

It is not hard to show that $\zeta(z)$ is actually the well-known "natural" solution (positive Taylor coefficients, cf. equation (4.3), [**MM89**]) of $zU(s) - s = 0$. $\zeta(z)$ is therefore analytic in the indented disk $\Delta_\rho = \Delta_\rho(\eta, \varphi) = \{z : |z| \leq \rho + \eta, |\arg(z - \rho)| \geq \varphi, z \neq \rho\}$ for some $\eta > 0$, $0 < \varphi < \pi/2$, and has only a single algebraic singularity of square-root type at $z = \rho$ on its circle of convergence.

In summary, Lemmas 4.1–4.4 of [**S94**] establish that for $0 < \alpha < \rho + v$, $v > 0$ sufficiently small there is some $\tau < r_\alpha < R_U$ such that $F(s,z) = 0$ has exactly two solutions $\zeta(z)$, $\kappa(z)$ (formed by the two analytic branches of a single double valued solution, hence "joining" at $\alpha = \rho$), which lie entirely in $\overline{\mathcal{D}}(0, r_\alpha)$ for $z \in \mathcal{D}(\alpha, \varepsilon)$ for ε sufficiently small. This, however, is exactly the information required for singularity analysis: The next lemma provides a uniform asymptotic expansion for $g_n(z) = [s^n]G(s,z)$ for $n \to \infty$, $z \in \mathcal{D}(\alpha, \varepsilon)$, where $G(s,z)$ denotes a function analytic in a neighborhood of $s = 0$ and $z = \alpha$. Note that —by virtue of well-known theorems from the theory of analytic functions of two complex variables, cf. [**Mar65**, p. 101ff], for example— $g_n(z)$ is *analytic* in a neighborhood of $z = \alpha$.

LEMMA 7.4. (cf. [**S94**, Lemma 4.5]) *Let $0 < \alpha < \infty$ be arbitrary but fixed. Suppose that $U(s)$ and $W(s)$ are analytic within the open disk $\mathcal{D}(0, R_U)$ and that there exists some r_α, $0 < r_\alpha < R_U$ such that*

$$G(s,z) = \frac{W(s)}{zU(s) - s}$$

has at most two simple poles $s = \zeta(z)$ and $s = \kappa(z)$, lying entirely in the interior of the closed disk $s \in \overline{\mathcal{D}}(0, r_\alpha)$ for every $z \in \mathcal{D}(\alpha, \varepsilon)$, $\varepsilon > 0$ sufficiently small. Then, $g_n(z) = [s^n]G(s,z)$ is analytic and fulfills

$$g_n(z) = \frac{W(\zeta(z))}{1 - zU'(\zeta(z))}\zeta(z)^{-(n+1)} + \frac{W(\kappa(z))}{1 - zU'(\kappa(z))}\kappa(z)^{-(n+1)} + O(r_\alpha^{-n})$$

for $n \to \infty$, where the remainder term denotes an analytic function and is uniform for $z \in \mathcal{D}(\alpha, \varepsilon)$. ∎

Due to the fact that $zU(s)-s$ has two simple poles for $z \neq \rho$ but a double one for $z=\rho$, Lemma 4.5 is not directly applicable in a neighborhood of ρ. However, it is possible to provide the appropriate asymptotics also in this case, cf. [**S94**, Lemma 4.6].

With the help of Lemma 7.4, it is not too difficult to obtain a recursive uniform asymptotic expansion of the desired $B^{(L)}_{T_L}(z)$ for $T_1,\ldots,T_L \to \infty$ in terms of the associated zeroes, which may in turn be fully characterized by closely investigating the functional equation $zP_L\big(B^{(L-1)}_{T_{L-1}}(s)\big) - s = 0$ arising in the denominator of (7.2). Solving the recursion, an asymptotic expression for $B^{(L)}_{T_L}(z)$ is obtained, which is uniform for $z \in \mathcal{D}(1,\varepsilon)$ and is hence differentiable. Substituting $z=1$, our major theorem follows:

THEOREM 7.5. (cf. [**S94**, Theorem 4.18]) *With the notations and conditions mentioned above, the successful run duration $\mathcal{S}_T$ for static priority scheduling with $L \geq 1$ priority levels is approximately exponentially distributed with parameter $1/\mu^{(L)}_{T_L,\ldots,T_1}$, where*

$$\mu^{(L)}_{T_L,\ldots,T_1} = \frac{1}{\sum_{\ell\in\mathcal{I}} d_\ell P_\ell(\kappa_\ell)^{-(T_\ell-1)}}\left(1+\frac{\sum_{i=1}^{L} O\big(r_i^{-T_i}\big)}{\sum_{\ell\in\mathcal{I}} O\big(P_\ell(\kappa_\ell)^{-T_\ell}\big)}\right)$$

for sufficiently large $T_1,\ldots,T_L \to \infty$ such that $T_\ell = O\big(T^{k_\ell}\big)$ for arbitrary but fixed k_ℓ and $T \to \infty$.

$\mathcal{I}$ is the set defined by $1 \in \mathcal{I}$ and $\ell \in \mathcal{I}$ for $2 \leq \ell \leq L$ if either the radius of convergence R_{P_ℓ} of $P_\ell(z)$ is less or equal to $\tau^{(\ell-1)}$, or $\tau^{(\ell-1)}/P^{(\ell)}(\tau^{(\ell-1)}) < 1$ otherwise; $\tau^{(\ell)}$ denotes the minimal positive solution of $xP^{(\ell)\prime}(x) - P^{(\ell)}(x) = 0$ for $P^{(\ell)}(z) = \prod_{j=1}^{\ell} P_j(z)$. Moreover, with κ_ℓ denoting the minimal solution of $x = P^{(\ell)}(x)$ for $x > 1$,

$$d_\ell = \frac{\big(1-P^{(\ell)\prime}(1)\big)^2}{1-P^{(\ell-1)\prime}(1)} \cdot \frac{1-P_\ell(\kappa_\ell)P^{(\ell-1)\prime}(\kappa_\ell)}{P^{(\ell)\prime}(\kappa_\ell)-1}\left(\frac{\kappa_\ell-1}{\kappa_\ell}\right)$$

where $P^{(0)}(z) \equiv 1$, and

$$r_\ell = \begin{cases} P_\ell(\kappa_\ell)+\epsilon & \text{for some } \epsilon>0, \text{ if } \ell \in \mathcal{I}, \\ \rho^{(L-1)}, & \text{if } \alpha_l = \tau^{(\ell-1)}/P^{(\ell)}(\tau^{(\ell-1)}) > 1 \text{ or } \alpha_l = 1 \text{ and all } T_1,\ldots,T_L \text{ finite}, \\ \rho^{(L-1)}-\epsilon & \text{for some } \epsilon>0, \text{ if } \alpha_l = 1 \text{ and arbitrary } T_1,\ldots,T_L. \end{cases}$$ ∎

Finally note that we also showed that, for finite T_L, there is a polar singularity $\xi^{(L)}_{T_L}$ at on the circle of convergence of $B^{(L)}_{T_L}(z)$. Actually, an expansion of $B^{(L)}_{T_L}(z)$ near $\xi^{(L)}_{T_L}$ is provided, which explains the somewhat puzzling limiting behavior as $T_L \to \infty$; cf. [**S94**, Lemma 4.14] for details.

8. CONCLUSIONS

We demonstrated the power of methods from the analysis of algorithms and data structures in the investigation of deadline meeting properties of scheduling algorithms for probabilistic aperiodic tasks in real-time systems. Contrasting the usual queueing theory devices,

our approach is based on exploiting combinatorial and asymptotic properties of certain random trees and works without any equilibrium assumption.

The comparison of the results for different scheduling techniques in the no-priority case confirms the expected superior performance of FCFS scheduling. Due to our fixed deadline assumption, the latter is equivalent to the *earliest deadline first* algorithm, which is known to be optimal, cf. [**SG91**]. Actually, there is a significant difference between deadline meeting properties of FCFS and LCFS scheduling in the normal case, since κ is always (usually much) larger than ρ. Preemptive and nonpreemptive LCFS scheduling perform roughly equivalent. Note that this ranking of our disciplines confirms the one obtained by steady-state results, cf. [**Coh82**, p. 475]. Our results, however, extend that qualitative ranking insofar that we quantify how much better some particular scheduling discipline performs over another one.

Needless to say, there are many important problems left for further research:

(1) Application of our approach to other scheduling algorithms, in particular to earliest deadline first in case of several different deadlines, and schemes designed for aperiodic task scheduling in conjunction with cyclic task schedulers, see [**LR92**], for example.

(2) Considering tasks with (additional) resource conflicts. In our approach, it was assumed that all tasks are competing for the server, but are otherwise independent. Dealing with tasks which may block each other when accessing additional mutually exclusive resources should be considered to meet practical needs.

(3) Provisions for time-variant and not independent arrival processes. This problem, pivotal to all attempts of analytic modelling of real applications, is not sufficiently solved by our approach. In order to preserve the tractability of involved computations, we essentially confine ourselves to Poisson arrivals, but it is questionable whether this is suitable for real-time applications. It should be mentioned that we are currently working on an elaborate monitoring system for distributed real-time systems (our *Versatile Timing Analyzer VTA*, see [**S94b**]), that will help us to obtain realistic information required to identify appropriate input models.

There are of course several other —and not at all trivial— improvements conceiveable, such as taking account of system overhead for scheduling and dispatching or relaxing the fixed deadline assumption.

Considering all aspects, however, we are convinced that our research develops in a quite promising direction: Experimental work devoted to measurement is needed to obtain appropriate system models, and refined techniques to deal with the theoretical part involved in the exploration of adequate models are also required. Evidently, most work remains to be done.

ACKNOWLEDGMENTS

We are greatful to J. Blieberger and A. Vrchoticky for reading and commenting on an earlier version of the manuscript. The overall appearance of the paper has been considerably improved by following W. Szpankowski's suggestion to include a relation of our

work to queueing theory research. Finally, K. M. Schossmaier's careful reading of the final version is also warlmy acknowledged.

References

AHS92. D. Aldous, M. Hofri, W. Szpankowski, *Maximum Size of a Dynamic Data Structure: Hashing with Lazy Deletion revisited*, SIAM J. Computing **21(4)** (August 1992), 713–732.

BBH84. F. Baccelli, B. Boyer, G. Hebuterne, *Single-Server Queues with Impatient Customers*, Adv. Appl. Prob. **16** (1984), 887–905.

Ben74. E.A. Bender, *Asymptotic methods in enumeration*, SIAM Review **16** (1974), 485–515.

BES93. J. Blazewicz, K. Ecker, G. Schmidt, J. Weglarz,, "Scheduling in Computer and Manufacturing Systems," Springer, 1993.

BKM91. S. Baruah, G. Koren, D. Mao, B. Mishra, A. Raghunathan, L. Rosier, D. Shasha, F. Wang, *On the Competitiveness of On-Line Real-Time Task Scheduling*, Proc. IEEE Real-Time Systems Symposium (December 1991), 106–115.

BS91. J. Blieberger, U. Schmid, *FCFS Scheduling in a Hard Real-Time Environment under Rush-Hour Conditions*, BIT **32** (1991), 370-383.

BS92. J. Blieberger, U. Schmid, *Preemptive LCFS Scheduling in Hard Real-Time Applications*, Performance Evaluation **15** (1992), 203–215.

Can84. E. R. Canfield, *Remarks on an Asymptotic Method in Combinatorics*, Journal of Combinatorial Theory, Series A **37** (1984), 348–352.

Coh82. J. W. Cohen, "The Single-Server Queue (rev. ed.)," North-Holland, Amsterdam, 1982.

CSB90. H. Chetto, M. Silly, T. Bouchentouf, *Dynamic Scheduling of Real-Time Tasks under Precendence Constraints*, Real-Time Systems **2** (1990), 181–194.

CSR88. S. Cheng, J. Stankovic, K. Ramamritham, *Scheduling Algorithms for Hard Real-Time Systems—A Brief Survey*, in "Tutorial: Hard Real-Time Systems," Computer Society Press IEEE, Washington, 1988.

Drm91. M. Drmota, *The Instability Time Distribution Behaviour of Slotted ALOHA*, Random Structures and Algorithms (Proc. Random Graphs '91) (to appear).

DLR82. M. A. H. Dempster, J. K. Lenstra, A. H. G. Rinnooy Kan (editors), "Deterministic and Stochastic Scheduling," D. Reidl Publishing Company, 1982.

DS93. M. Drmota, U. Schmid, *Exponential Limiting Distributions in Queueing Systems with Deadlines*, SIAM J. Appl. Math. **53(1)** (1993), 301–318.

DS93b. M. Drmota, U. Schmid, *The Analysis of the Expected Successful Operation Time of Slotted ALOHA*, IEEE J. Inf. Th. **39(5)** (1993), 1567–1577.

Fla79. 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.

FRV79. Ph. Flajolet, J. C. Raoult, J. Vuillemin, *The number of registers required to evaluate arithmetic expressions*, Theoret. Comput. Sci. **9** (1979), 99–125.

FO90. Ph. Flajolet, A. Odlyzko, *Singularity Analysis of Generating Functions*, SIAM J. Discr. Math. **3** (1990), 216–240.

GS77. B. Gavish, P. J. Schweitzer, *The Markovian Queue with Bounded Waiting Time*, Management Science **23(12)** (August 1977), 1349–1357.

KP87. P. Kirschenhofer, H. Prodinger, *On the Recursion Depth of Special Tree Traversal Algorithms*, Information and Computation **74** (No. 1, July 1987), 15–32.

Kle75. L. Kleinrock, "Queueing Systems," John Wiley and Sons, New York, 1975.

KS92. G. Koren, D. Shasha, *D^{over}: An Optimal On-Line Scheduling Algorithm for Overloaded Real-Time Systems*, Proc. IEEE Real-Time Systems Symposium, Phoenix, Arizona (December 1992), 290–299.

Kur93. J. Kurose, *Open Issues and Chellanges in Providing Quality of Service Guarantees in High-Speed Networks*, ACM Computer Communication Review **23(1)** (January 1993), 6–15.

KV91. C. M. Kenyon-Mathieu, J. S. Vitter, *The Maximum Size of Dynamic Data Structures*, SIAM J. Computing **20(5)** (October 1991), 807–823.

LKS91. G. Louchard, C. Kenyon, R. Schott, *Data Structures Maxima*, Proc. 8th Int. Conference on Fundamentals of Computing Theory FCT'91, Gosen, Germany (September 1991), 339–349.

LR92. J.P. Lehoczky, S. Ramos-Thuel, *An Optimal Algorithm for Scheduling Soft-Aperiodic Tasks in Fixed-Priority Preemptive Systems*, Proc. IEEE Real-Time Systems Symposium, Phoenix, Arizona (December 1992), 110–123.

Mai91. R. S. Maier, *Clolliding Stacks: A Large Deviations Analysis*, Random structures and Algorithms **2(4)** (1991), 379–420.

MS93. R. S. Maier, R. Schott, *The Exhaustion of Shared Memory: Stochastic Results*, Proc. 3rd Workshop Algorithms and Data Structures WADS'93, Montreal, Canada (August 1993), 495–505.

Mar65. M. Markushevic, "Theory of Functions of a Complex Variable," Prentice Hall, 1965.

MM78. A. Meir, J.W. Moon, *On the altitude of nodes in random trees*, Canad. J. Math. **30** (1978), 997–1015.

MM89. A. Meir, J. W. Moon, *On an Asymptotic Method in Enumeration*, Journal of Combinatorial Theory, Series A **51** (1989), 77-89.

P83. M. Pinedo, *Stochastic Scheduling with Release Dates and Due Dates*, Operations Research **31(3)** (May-June 1983), 559–572.

PTW88. S. S. Panwar, D. Towsley, J. K. Wolf, *Optimal Scheduling Policies for a Class of Queues with Customer Deadlines to the Beginning of Service*, JACM **35(4)** (October 1988), 832–844.

S94. U. Schmid, *Static Priority Scheduling of Aperiodic Real-Time Tasks*, (submitted) (1994).

S94b. U. Schmid, *Monitoring Distributed Real-Time Systems*, Real-Time Systems **7** (1994).

SB92. U. Schmid, J. Blieberger, *Some investigations on FCFS Scheduling in Hard Real-Time Applications*, J. Comput. Syst. Sci. **45** (1992), 493–512.

SB94. U. Schmid, J. Blieberger, *On nonpreemptive LCFS Scheduling with deadlines*, to appear in J. Algorithms (1994).

SG91. A. D. Stoyenko, L. Georgiadis, *An Optimal Lateness and Tardiness Scheduling in Real-Time Systems*, Compting **47** (1992), 215–234.

Ta77. L. Takács, "Combinatorial Methods in the Theory of Stochastic Processes," Robert E. Krieger Publishing Company, Huntington, New York, 1977.

TK91. A. M. van Tilborg, G. M. Koob (eds.), "Foundations of Real-Time Computing: Scheduling and Resource Management," Kluwer Academic Publishers, 1991.

Tow93. D. Towsley, *Providing Quality of Service in Packet Switched Networks*, Proc. Joint Conference Performance'93 and Sigmetrics'93, Lecture Notes on Computer Science **729** (1993), 560–586, Springer.

VF90. J. S. Vitter, Ph. Flajolet, *Average Case Analysis of Algorithms and Data Structures*, Handbook of Theoretical Computer Science (J. van Leeuwen, ed.) (1990), North Holland.

ZS89. W. Zhao, J. A. Stankovic, *Performance Analysis of FCFS and Improved FCFS Scheduling Algorithms for Dynamic Real-Time Computer Systems*, Proc. IEEE Real-Time Systems Symposium, Santa Monica, California (December 1989), 156–165.