NTNU
Norwegian University of
Science and Technology

Flow-times in an M/G/1 Queue under a Combined Preemptive/Non-preemptive Priority Discipline.

Scheduled Waiting Time on Single Track Railway Lines

Johan Narvestad Fatnes

Master of Science in Physics and Mathematics
Submission date: June 2010
Supervisor: Øyvind Bakke, MATH
Co-supervisor: Christine Handstanger, Jernbaneverket

Norwegian University of Science and Technology
Department of Mathematical Sciences

Problem Description

The Norwegian railway owner and operator, Jernbaneverket, proposed a priority based decision rule to be used during the process of scheduling trains operating on single track railway lines. The line segment to be studied only allows for one train to operate at a time. The scheduling rule works by favorizing trains with high priority on the line when operational requests come in conflict.

When studied under the assumption of only two priority classes, the scheduling rule was seen to have properties in common with both non-preemptive and preemptive repeat-different priority disciplines. Both disciplines are well documented in the literature.

The goal now is to generalize the scheduling problem to a system with an arbitrary number of train classes of different priority and then apply queuing theory to obtain an exact expression for the average scheduled waiting times.

Assignment given: 22. January 2010
Supervisor: Øyvind Bakke, MATH

Preface

This work was carried out at the Department of Mathematical Science at the Norwegian University of Science and Technology (NTNU) in spring 2010. The present report is a master thesis written under the course code TMA4905 (Statistics, Master Thesis) on request from the Norwegian railway operator and owner, Jernbaneverket. The thesis leads to the degree Master of Science and completes the 5-years Industrial Mathematics master program at NTNU.

I am most grateful for the guidance and constructive feedback I have received from my supervisor at the Department of Mathematical Science, associate professor Øyvind Bakke. His suggestions and support have been of great importance to the progress of my work.

Also, I would like to thank my co-supervisor at Jernbaneverket, Anne Christine Torp Handstanger, for posting an interesting problem and providing me with first hand insight on railway operations. She has also been very helpful pointing me in the right direction of useful literature on both queuing theory and railway scheduling.

Finally, I would like to thank my fellow students and friends for five wonderful years at NTNU. Studying would have been no fun without you.

Johan Narvestad Fatnes

Trondheim,
June 2010

Abstract

A priority based rule for use during the process of scheduling trains operating on a single track railway line was proposed by the Norwegian railway operator and owner, Jernbaneverket. The purpose of this study is to investigate the effect of the suggested scheduling rule on the scheduled waiting times suffered by trains operating on a segment of the railway line.

It is shown that the scheduling rule, under certain limiting assumptions, can be studied in the setting of queuing theory and that it has properties in common with a theoretical priority discipline combining two well documented priority rules. The main part of this study is the development and analysis of a threshold based, combined preemptive/non-preemptive priority discipline. Under the combined discipline, preemptions are allowed during the early stage of processing only. Theoretical expressions for flow-times of jobs passing through the queuing system are reached through detailed studies of the non-preemptive and the preemptive priority discipline.

The relationship between the suggested priority based scheduling rule and the theoretical, combined priority discipline is finally illustrated by simulations. When adjusted for actual time spent by trains on traversing the line segment, the steady state solution for flow-times obtained from queuing theory yields an accurate expression for the trains' average scheduled waiting times. The scheduling problem can in fact be modeled accurately by an $M/G/1$ queue under the combined priority discipline.

Contents

1 Introduction and scope

For a railway network the maximum capacity is an upper limit on the number of trains that can operate on a given part of the railway network in a given period of time. Several factors contribute to reduce the maximum capacity. In practice the capacity is dependent on the types of trains operating on the lines – the rolling stock. The trains' speed, acceleration, stop length and stopping pattern put restraints on the effective capacity. In addition there must be some time buffer between trains so that unexpected delays do not transfer to the other trains.

Since all these limitations must be incorporated in the final schedule it follows that scheduling is a very important part of managing a railway network. It makes the train traffic predictable, produces data for timetables essential for passengers and it is of great importance for traffic control and safe operation in the railway network.

This study is concerned with the basics of the scheduling procedure. The train operators request to operate their train on a given train path; a railway line allocated to a train in a given period of time. Often such requests for paths come in conflict with each other. Two train operators may prefer to use the same part of the infrastructure at the same time.

It is crucial that the infrastructure owner avoids these conflicts when assigning train paths to the train operators. The result is less favorable train paths being assigned to those trains and operators that have low priority on the line in question. Consequently, the final authorized times for operation may differ from the originally requested times. This difference between the preferred time and the final, scheduled time for operation is called scheduled waiting time. Scheduled waiting time is, as the name suggests, incorporated in the final schedule.

It is important not to confuse scheduled waiting time with unscheduled waiting time. Unlike scheduled waiting time, the latter is often due to technical errors or human failure and affects passengers and goods transporters in form of unpredicted delays and changes in the planned schedule.

The coordination of train paths in order to optimize the usage of the existing infrastructure is the underlying problem of this work. The goal of the Norwegian infrastructure owner, Jernbaneverket, is to offer the train operators best possible terms for their operations. The scheduled waiting time should be minimized under given criteria.

As of today, passenger traffic has higher priority than freight traffic in the Norwegian railway network. This situation may very well change in the future. A large amount of goods is transported by road since this actually provides a more flexible and often faster alternative to freight trains. Giving freight trains higher priority is a way of making the railway more attractive to the goods transporters and may create a possibility for the infrastructure owner to price the train paths higher and thus earn more money.

A new decision rule for use under the scheduling process is suggested by Jernbaneverket. It sets new criteria for how trains of various priority should be treated when conflicts in requested operation times arise. This work explore how this new priority rule affects the scheduled waiting times.

The Norwegian railway network mainly consists of single track railway lines. Compared to double track lines, this causes extra scheduled waiting time in situations where different train operators want to travel the same line in opposing directions at more or less the same time. As double track railway lines are common in most of Europe, the theoretical treatment of single track lines is somewhat limited.

This study is limited to scheduling in a single block section. A block section is defined to be the line segment located between two main signals. For safety reasons only one train can operate in such a block section at a time. More details regarding railway infrastructure is given by Hansen and Pachl (2008), Handstanger (2009) and Skartsæterhagen (1993).

It is necessary to consider the total elapsed time a section of track is allocated exclusively to a train movement and therefore blocked for other trains. This is blocking time, by definition. It starts when a train is given authority to move into the block section and lasts until the train completely has left the section. Blocking time consists of several independent stochastic parts. However, the main part of it is simply the time the actual train needs to traverse the block section in question. Here it is assumed that the blocking time is the same for all trains of a given type and solely dependent on their speed. The fact that part of the blocking time is random is neglected.

***

The scheduling rule proposed by Jernbaneverket is based on trains being classified into different priority classes. Each class has a predefined level of priority which can depend on factors such as train type, importance of operation or traveling direction.

The train operators request to enter a block section at a given time. For a train from a class of low priority to be allowed to operate in the block section at the requested time, there must be enough time for it to traverse a predefined fraction φ of that block section before a train of higher priority is requested to operate. The threshold value φ defines the effect of the priority rule. If the time criterion is not fulfilled, the train of low priority must wait. The train of high priority is then given authority to operate in the block section at the requested time.

If the time criterion is fulfilled, it is the train of high priority that must wait while the train of low priority completes its operation. There might be several high priority requests placed for the time interval the line is allocated to the train of lower priority. Once the low priority train is scheduled to exit the block section, the train of highest priority, scheduled to wait, is granted authority to enter the section. In case of more than one train from the same

priority class being scheduled to wait, the train with the earliest request is authorized to operate first. That is, a first-come-first-served principle is practiced within each priority class.

The main question to address is what impact this scheduling rule will have on the scheduled waiting times. Under a set of assumptions, accounted for in Section 8, the problem can be studied in the setting of queuing theory.

In a pre-project (Fatnes, 2009) a simplified model with only two priority classes of trains was studied. From simulations it was shown that average scheduled waiting time suffered due to the above scheduling rule lies between the mean waiting time in two well documented priority systems (Figure 1). A threshold value $\varphi = 1$ was seen to yield a strictly preemptive queuing system, while $\varphi = 0$ resulted in a strictly non-preemptive one. In the situation with two priority classes it was shown that the average scheduled waiting times were somewhere between the mean waiting times in a non-preemptive priority queue and the mean waiting times in a preemptive priority queue. Low threshold values were seen to give a system with high overall traffic capacity. High φ-values were seen to have a positive effect on the scheduled waiting times for trains of high priority, but reducing the effective capacity of the system.

In Figure 1, class 1 trains have priority over class 2 trains. λ_1 and λ_2 are the average number of operations requested per minute for each of the two priority classes. P_1 and P_2 denotes the corresponding blocking times. Further details are not important at this point.

The purpose of this study is to investigate the effect of the suggested, priority based, scheduling rule on average scheduled waiting times closer. The scheduling problem is generalized to an arbitrary number of priority classes. Queuing theory is then applied and the properties of the non-preemptive and the preemptive priority disciplines are analyzed. It will be shown that exact expressions for average scheduled waiting times in the scheduling problem can be obtained by considering a combination of the two priority disciplines.

***

The scheduling problem is studied in the setting of queuing theory and the main part of this work evolves around various forms of priority queues. Before priority queues are introduced some of the basic concepts in the field of queuing theory are considered. Section 2 contains a brief discussion of fundamental ideas together with a short introduction to the notation used throughout the study. The queuing system itself is defined and the interaction between arriving jobs and a processing unit is explained.

The next two sections deal with busy periods and delay cycles, the latter being a generalization of the first. A simple queuing system with only one type of jobs is considered. A busy period describes a time interval in which the queuing systems' processing unit is busy processing jobs. The results obtained for lengths of busy periods and delay cycles in Section 3

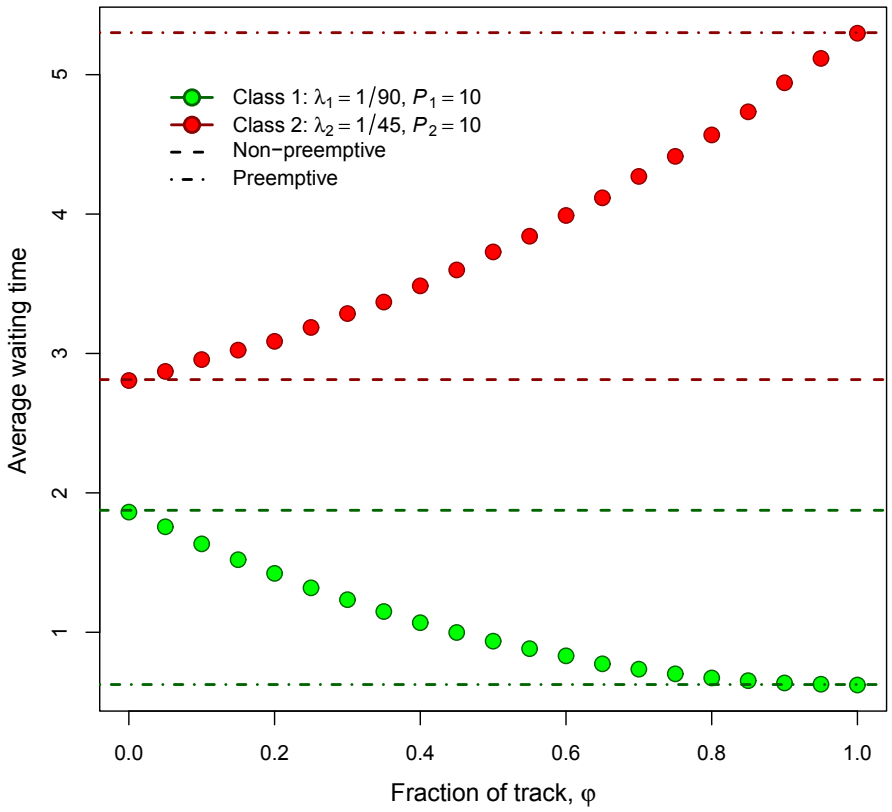


Figure 1: Results from a previous study of a scheduling scenario with only two priority classes.

and Section 4, respectively, are fundamental building blocks for the further exploration of priority queuing systems.

In Section 5 a priority system is introduced. The arriving jobs are divided into different priority classes and the basic, non-preemptive priority system is considered. It is characterized by waiting jobs belonging to the class of highest priority being processed first. By studying various delay cycles the expected time a job from a given priority class spends in the system (flow-time) is found.

In the next section, Section 6, a more advanced priority discipline is considered. Under the preemptive priority discipline studied here, jobs of high priority are allowed to interrupt the processing of less important jobs. The computation of flow-times becomes more complex in this case.

In Section 7 the core result of this study is obtained: Equation (43)

states the expected flow-time of a job in a queuing system under a combined preemptive/non-preemptive priority discipline. The priority system is characterized by preemptions of low prioritized jobs being allowed only during the early stage of the processing. Once a certain proportion φ of the total processing time has been completed, the remaining processing time is protected from preemptions.

When focus is shifted back to the scheduling problem in Section 8 it is argued that average scheduled waiting times for trains can be found from the expected flow-times in the combined preemptive/non-preemptive priority system when adjusted down by the actual trains' blocking time. A scheduling scenario is simulated, confirming the theoretical result obtained from queuing theory. The source code used for the simulations is presented in Appendix C.

Appendix A and Appendix B contain brief, but important, introductions to theory and concepts crucial to the rest of the study. Results from both transform theory (Appendix A) and renewal theory (Appendix B) are referred to throughout the text, but since these topics are not directly connected to the queuing theory they are kept separated from the main part.

2 Preliminary queuing theory

Before one can study the details of specific queuing system some general queuing theory and basic queuing models must be considered. This section deals shortly with the basic theory and concepts.

Ross (2007) discusses probability models and properties of stochastic processes, including the very important Poisson process. Basic queuing theory is very well presented by Kleinrock (1975). The notation in this study is chosen to fit the more advanced work of Conway et al. (1967) and Drekic and Stanford (2000) on priority queues.

2.1 The foundation of queuing theory

A basic queuing model assumes that jobs arrive at the queuing system by an arrival process. A service unit is responsible for processing the jobs. In general it is possible to have a system with several service units, but the concern of this text is the single server system. The entire queuing system considered consists of a queue of waiting jobs and a machine that processes one job at a time. Once a job is processed completely it departs from the system and is no longer of any concern, neither to the machine nor the other jobs waiting in queue for processing.

The number of jobs in the system is the number of jobs waiting in queue plus the job currently being processed. It is common in the literature to denote this quantity by L and that convention is followed here as well.

In order to define the system, rules for how jobs in the queue are selected for processing – the selection discipline – and the distribution of the processing times have to be specified. The selection discipline can be a simple first-come-first-served (FCFS) discipline where the job first in line gets served first, or it can be specified by sophisticated rules depending on job classes and state of the system. In many situations a need for some sort of priority for important jobs arises.

The main interest of this text is the so-called $M/G/1$ queuing system under different priority selection disciplines.

In the queuing literature it is common to describe the queuing system by the Kendall notation $A/B/n/m$. A denotes the type of arrival process, B denotes the type of processing process, n is the number of service units, or machines, processing jobs and m is the size of the waiting space. When there is no limit for the queue length, $m = \infty$ is often omitted from the notation. So also in the following.

In the $M/G/1$ queuing system the arrival process is Markovian, the processing is according to some arbitrary process and there is only one machine processing jobs. This really means that arrivals of jobs at the system are according to a Poisson process, say with an arrival rate λ, and that the processing time distribution is some general probability distribution.

Assume job n ($n = 1, 2, \dots$) arrives at the system at time τ_n where

$$0 < \tau_1 < \tau_2 < \cdots \quad (n = 1, 2 \dots).$$

The difference between succeeding arrival times, called inter-arrival times, is defined as

$$t_n = \tau_n - \tau_{n-1} \quad \text{for } n = 1, 2, \dots$$

where $\tau_0 = 0$ by definition.

The inter-arrival times can often be assumed to be independent realisations of a random variable T_A with cumulative probability distribution given by

$$\Pr(T_\mathrm{A} \leq t) = A(t).$$

Assume that the number of arrivals N, in a fixed time interval of length t, can be modeled by a Poisson process[1]

$$\Pr(N = n) = \frac{(\lambda t)^n e^{-\lambda t}}{n!},$$

where the arrival rate λ is the expected number of arrivals per unit time. If the number of arrivals follows the above Poisson process with an average of λt arrivals in the time interval $[0, t]$, the inter-arrival times t_n, $n = 1, 2, \dots,$

[1] N denotes different quantities throughout the text. Its meaning should be clear from the context.

is independently exponentially distributed with mean $1/\lambda$ according to Ross (2007, page 307–308)

$$A(t) = 1 - e^{-\lambda t}.$$

The exponential distribution possesses the Markovian property, it is memoryless. Mathematically that is expressed as

$$\Pr\left(T_A > t + s \mid T_A > s\right) = \Pr\left(T_A > s\right).$$

In words this statement says that the probability of an inter-arrival time lasting for at least $t + s$ time units, given that it has already been s time units since the last arrival, equals the initial probability for an inter-arrival time to last for at least t time units. Because of this property the Poisson arrival process is referred to as a Markovian process, and hence the M in the Kendall notation $M/G/1$.

There are two more properties of the exponential distribution that are of great importance in queuing theory. Assume that $T_1, \ldots, T_K$ are independent exponentially distributed random variables with respective means $1/\lambda_1, \ldots, 1/\lambda_K$. From

$$\Pr\left(\min\{T_1, \ldots, T_K\} > t\right) = \prod_{k=1}^{K} \Pr\left(T_k > t\right) = e^{-t\sum_{k=1}^{K}\lambda_k}$$

it is clear that $\min\{T_1, \ldots, T_K\}$ is exponentially distributed with parameter $\lambda_1 + \cdots + \lambda_K$. Furthermore, it can be shown that

$$\Pr\left(T_k = \min\{T_1, \ldots, T_K\}\right) = \frac{\lambda_k}{\lambda_1 + \cdots + \lambda_K}$$

(Ross, 2007, page 294). In a system with arrivals occuring according to K independent Poisson arrival processes with arrival rates $\lambda_1, \ldots, \lambda_K$, respectively, the above expression state the probability for the next arrival to be from the kth process having arrival rate λ_k. Finally, also explained by Ross (2007, page 70), a combination of independent Poisson processes is a new Poisson process with rate given as the sum of the rates of the combined processes. These properties become very important in the discussion of priority systems, since the arriving jobs are divided into priority classes with possibly different arrival rates.

The processing time of the nth arriving job is denoted P_n. These processing times are assumed to be independent realisations of a random variable P, distributed according to a general probability distribution $G(p)$, and thus the G in the Kendall notation $M/G/1$ explains itself.

To study the $M/G/1$ queuing system it is convenient to define some additional terms and quantities. First, let $U(t)$ be a function describing the amount of unfinished work, or the remaining time necessary to process all customers currently present in the system, at time t.

Second, it is necessary to distinguish between a busy and an idle system. While the machine is processing a job the system is said to be busy. When there is no job present for the machine to process the system is said to be idle. In the case of a standard $M/G/1$ queuing system the system is busy as long as there is one or more jobs present. The duration of such a busy period will be denoted by T.

It has been shown that the proportion of time the system is busy in the long run, is the average job arrival rate times the average time a job keeps the machine busy (Kleinrock, 1975, page 18). This quantity is called the system utilization factor and denoted by ρ. Under both the FCFS discipline and the non-preemptive priority discipline, the latter considered in Section 5, the selection discipline does not affect the total processing time of any job. Neither is the unfinished work function $U(t)$ effected by these selection disciplines. Systems with this property are said to be work-conserving (Wolff, 1970, page 327). It follows that the average time a job keeps the machine busy is the average processing time $\mathrm{E}[P]$. Thus the utilization factor is given as

$$\rho = \lambda \mathrm{E}[P]$$

in a work conserving system. Note also that the proportion of time the system is idle in the long run now is given as $1-\rho$. In general the utilization factor is the ratio of the rate at which work enters the system to the rate at which the system can perform this work.

In all that follows $\rho < 1$. Queuing systems with this property is said to be non-saturated, and the restriction on ρ is necessary to avoid a queue with length growing to infinity with time. In the case of a saturated system the jobs arrive faster than the machine can process them and the theory developed in the next few sections will break down.

The main focus in the theoretical part of this text is the expected flow-time for a job passing through a queuing system. Flow-time F is defined by Conway et al. (1967, page 11) as the total amount of time a job spends in the system.

Also the waiting time W is of interest. For a given job the waiting time is defined as time spent in the queue from the time of arrival to the time the processing starts. Hence, the relationship between flow-time and waiting time is

$$F = W + P.$$

As a final remark to the general queuing theory, once the average time a job spends in the system is known, Little's formula (Little, 1961, page 383) can be used to obtain the average number of jobs in the system. Little proved under very general assumptions that the average number of jobs in a queuing system is the average arrival rate of jobs to the system times the average time a job spends in the system. That is

$$\mathrm{E}[L] = \lambda \mathrm{E}[F],$$

where $\mathrm{E}[L]$ is the average number of jobs present in the system in the long run.

2.2 Notation

Throughout the text various random variables are encountered. A continuous random variable X is distributed according to the probability distribution $F_X(x) = \Pr(X \leq x)$. Its probability density function is $f_X(x) = \mathrm{d}F_X(x)/\mathrm{d}x$.

Laplace transforms associated with probability density functions are used to obtain moments of the corresponding random variables through the following relationship: For a continuous random variable X, distributed according to the probability density function $f_X(x)$, the Laplace transform is given by

$$\phi_X(z) = \int_{-\infty}^{\infty} e^{-zx} f_X(x)\ \mathrm{d}x = \mathrm{E}[e^{-zX}],$$

where z is a complex variable.

The Laplace transform is simply the moment generating function evaluated at $-z$. It is convenient to consider the Laplace transform rather than the moment generating function of X itself, as the Laplace transform always is between 0 and 1 when the random variable is non-negative and $z \geq 0$ (Ross, 2007, page 72). In queuing theory the random variables considered describe time intervals and are in fact non-negative.

Now, by the close relationship between Laplace transforms and moment generating functions, by evaluating the ith derivative of $\phi_X(z)$ as z goes to 0, the ith moment of X can be found as

$$\mathrm{E}\left[X^i\right] = \lim_{z\to 0}(-1)^i \frac{\mathrm{d}^i \phi_X(z)}{\mathrm{d}z^i}.$$

For more details on the and a brief discussion of the properties of the Laplace transform, see Appendix A. To get a somewhat more compact notation the following simplification is introduced:

$$\phi_X^{(i)}(0) = \lim_{z\to 0} \frac{\mathrm{d}^i \phi_X(z)}{\mathrm{d}z^i}.$$

Throughout the text the generic notation presented above will be used for most of the random variables encountered. An exception is made for the processing time, P. As mentioned the processing times are distributed according to the general distribution $G(p) = \Pr(P \leq p)$ and has probability density function $g(p) = \mathrm{d}G(p)/\mathrm{d}p$. In addition the corresponding Laplace transform is denoted by $\gamma(z)$.

When the priority rules becomes more sophisticated it will be necessary to consider different versions of processing time. These differences will be

denoted by subscripts on the variables and their corresponding distributions, density functions and Laplace transforms.

Another exception is made for the duration T of a busy period. This random variable is distributed according to the distribution given by $H(y) = \Pr(T \leq y)$ and its probability density function and Laplace transform are denoted $h(y)$ and $\eta(z)$, respectively.

Further exceptions may occur in places where the readability of the text is increases significantly by shorter notation.

3 Busy periods

The standard $M/G/1$ queuing system experiences alternating cycles of busy and idle periods. By studying the distribution of the duration of such busy periods, one arrives at a result which in turn can be used to study the distribution of flow-time.

Theory regarding busy periods is given by Kleinrock (1975, Chapter 5.8), Conway et al. (1967, Chapter 8-3) and Takagi (1991, Chapter 1.2).

3.1 The length of busy periods

Recall from Section 2 that during an idle period there are no jobs present in the queuing system. The idle period terminates at the time of an arrival at the system. In a busy periods there is at least one job in the system which demands attention and keeps the machine busy. When the system is emptied of all jobs, a new idle period is initiated (Ross, 2007, page 530).

Recall also that job n arrives at the system at time τ_n and that $t_n = \tau_n - \tau_{n-1}$ is the inter-arrival time between job $n-1$ and job n. Let P_n denote the processing time of job n. Furthermore, $U(t)$ is the amount of unfinished work in the system – or the remaining time required to process all customers present in the system – at time t.

Now, denote the duration of the busy periods as $T_1, T_2, \ldots$ and the duration of the idle periods as $I_1, I_2, \ldots$. Consider an empty system and assume that job 1 arrives at time τ_1, enters the machine for processing, immediately initiating a busy period. The unfinished work in the system was clearly zero prior to the arrival of this job. At time $t = \tau_1$ the amount of unfinished work in the system jumps to P_1. $U(t)$ decreases at rate -1 until the arrival of job 2 causes it to make a vertical jump of magnitude P_2 at time $t = \tau_2$, and so on. The busy period initiated by job 1 lasts until the system once again is emptied of all work.

Since the arrival process is memoryless the calculation of the idle period distribution is trivial. An idle period terminates at an arrival. The I_n's are thus independently distributed according to the same distribution as the inter-arrival times. The distribution of I_n is simply $\Pr(I \leq t) = 1 - e^{-\lambda t}$ for all $n = 1, 2, \ldots$.

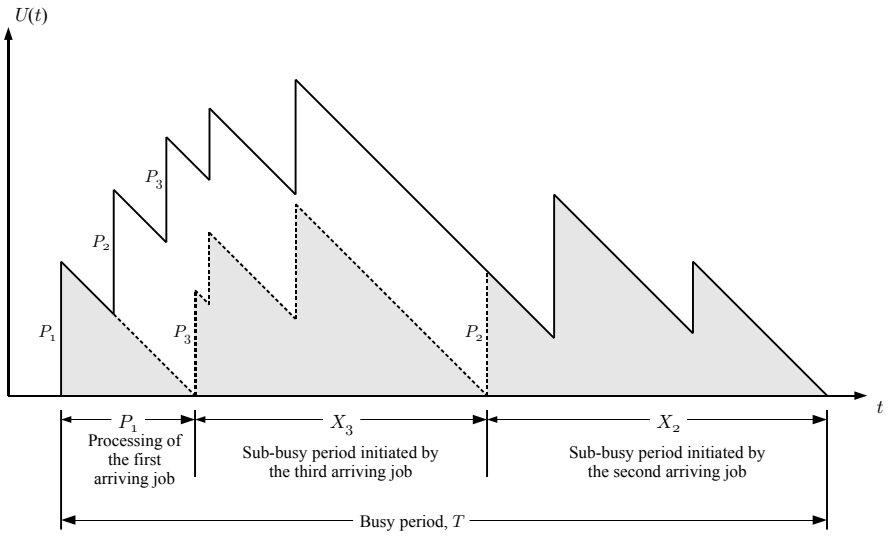


Figure 2: The development of unfinished work $U(t)$ and sub-busy periods.

It should be noted that the order in which the jobs are processed does not affect the graph, as long as the selection discipline does not imply any extra processing time. To obtain the distribution $H(t) = \Pr(T \leq t)$ of the busy period duration T, one can take advantage of this observation. The calculation is based on a selection discipline for which the busy period can be viewed as the processing time of the initiating job plus a series of random variables which them self are distributed as the length of a busy period (Conway et al., 1967, page 149).

The busy period is viewed in terms of sub-busy periods, illustrated by an example in Figure 2 (Kleinrock, 1975, page 209). The idea is to assume that two different queues of waiting jobs are formed and that the selection discipline in each queue is last-come, first-served (LCFS). All jobs arriving during the processing time P_1 enter the initial queue. All jobs arriving at the system after P_1 enter the main queue. A job is always selected from the main queue if possible. If the main queue is empty jobs are selected from the initial queue. The busy period terminates when both queues are emptied. Figure 2 shows a situations where two jobs arrives at the system during the processing time of the initiating job.

A job from the initial queue is selected for processing only if the main queue is empty. While this job is being processed other jobs may arrive and these enter the main queue. The jobs in the main queue are then selected for

processing, starting with the one that arrived last, until the main queue once again is empty. Now, and this is the crucial observation, the time period between two successive jobs from the initial queue – a sub-busy period – has all the characteristics of a busy period.

Furthermore, the sub-busy periods have lengths that are independently and identically distributed according to the same distribution as the real busy period. This is discussed both by Kleinrock (1975, page 211) and by Conway et al. (1967, page 150).

The duration of a busy period T is the sum of $N+1$ random variables, where N is the number of jobs arriving during P_1, the processing time of the job initiating the busy period. Thus

$$T = P_1 + X_{N+1} + \cdots + X_3 + X_2,$$

where X_i is the sub-busy period generated by the ith arriving job, as illustrated in Figure 2.

The Laplace transform of the probability density function $h(t)$ of T is

$$\eta(z) = \int_{t=0}^{\infty} e^{-zt} \, \mathrm{d}H(t) = \mathrm{E}\left[e^{-zT}\right].$$

Conditioning on the the processing time of job 1 and the number of jobs arriving during this processing time, N, one can write

$$\begin{aligned}
\mathrm{E}\left[e^{-zT} \mid P_1 = p,\, N = n\right] &= \mathrm{E}\left[e^{-z(p+X_{n+1}+\cdots+X_2)}\right] \\
&= \mathrm{E}\left[e^{-zp}\right] \mathrm{E}\left[e^{-zX_{n+1}}\right] \cdots \mathrm{E}\left[e^{-zX_2}\right] \\
&= e^{-zp}[\eta(z)]^n.
\end{aligned}$$

The second equality follows from the fact that the duration of the sub-busy periods are independent. The last equality holds since P_1 is given and all sub-busy periods are identically distributed with Laplace transforms $\eta(z)$.

N is Poisson distributed with mean λp since it represents the number of arriving jobs in a time interval of length p. Thus the condition on N can be avoided by application of the law of total probability. From the equation above

$$\begin{aligned}
\mathrm{E}\left[e^{-zT} \mid P_1 = p\right] &= \sum_{n=0}^{\infty} \mathrm{E}\left[e^{-zT} \mid P_1 = p,\, N = n\right] \Pr\left(N = n\right) \\
&= \sum_{n=0}^{\infty} e^{-zp}[\eta(z)]^n \frac{(\lambda p)^n}{n!} e^{-\lambda p} \\
&= e^{-p[z+\lambda-\lambda\eta(z)]}.
\end{aligned}$$

Further, integrating with respect to the processing time distribution $G(p)$ yields

$$\eta(z) = \int_{p=0}^{\infty} e^{-p[z+\lambda-\lambda\eta(z)]} \, \mathrm{d}G(p).$$

Finally, the right-hand side of this is recognized as the transform of $G(p)$ evaluated at $z + \lambda - \lambda\eta(z)$ and an important relationship,

$$\eta(z) = \gamma[z + \lambda - \lambda\eta(z)], \tag{1}$$

has been established. This functional equation is usually impossible to solve, but it can be used to obtain the moments of the busy period duration.

The ith moment of the busy period duration T is

$$\mathrm{E}\left[T^i\right] = (-1)^i \eta^{(i)}(0)$$

and similarly, the ith moment of processing time is

$$\mathrm{E}\left[P^i\right] = (-1)^i \gamma^{(i)}(0).$$

From Equation (1)

$$\begin{aligned}\mathrm{E}\left[T\right] &= -\lim_{z\to 0} \frac{\mathrm{d}\gamma[z + \lambda - \lambda\eta(z)]}{\mathrm{d}z} \\ &= -\gamma'(0)\left[1 - \eta'(0)\right] \\ &= \mathrm{E}\left[P\right]\left(1 + \mathrm{E}\left[T\right]\right).\end{aligned}$$

Used here is the fact that $z + \lambda - \lambda\eta(z) \to 0$ when $z \to 0$ since

$$\eta(0) = \int_{t=0}^{\infty} \mathrm{d}H(t) = 1.$$

Solving for $\mathrm{E}\left[T\right]$ yields

$$\mathrm{E}\left[T\right] = \frac{\mathrm{E}\left[P\right]}{1 - \lambda\mathrm{E}\left[P\right]} = \frac{\mathrm{E}\left[P\right]}{1 - \rho}, \tag{2}$$

where $\rho = \lambda\mathrm{E}\left[P\right]$ is the utilization factor of the system introduced in Section 2.

The second moment is in a similar way found to be

$$\mathrm{E}\left[T^2\right] = \frac{\mathrm{E}\left[P^2\right]}{(1 - \rho)^3}$$

(Kleinrock, 1975, page 214).

3.2 Flow-time in a first-come-first-served system

In the following the knowledge of the duration of the busy period developed above is used to calculate the expected flow-time of a job in a FCFS queuing system. The flow-time of a job is defined as the entire time between arrival at the system and process completion. Flow-time, denoted by F, consists in

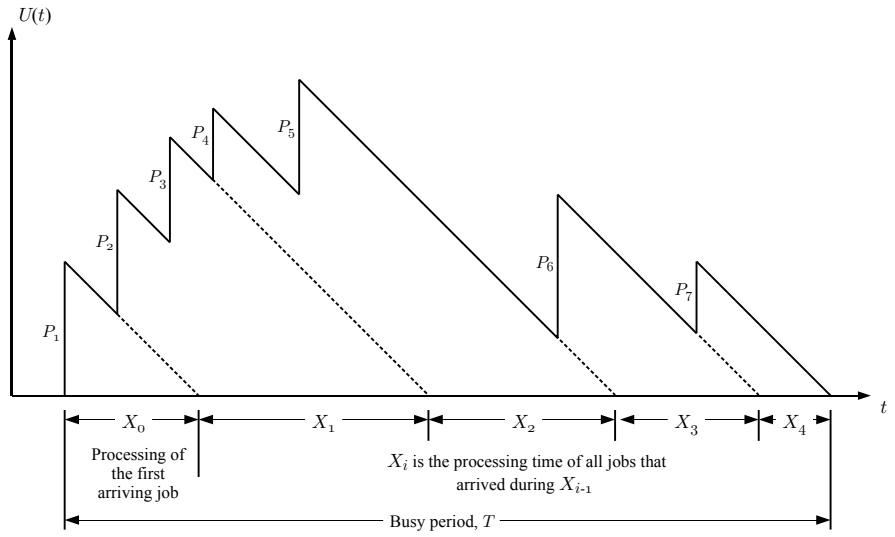


Figure 3: The first-come-first-served decomposition of a busy period.

other words of both waiting time and processing time. The corresponding Laplace transform is denoted $\phi_F(z)$.

Jobs that arrive while the machine is idle will have flow-time equal their processing time. Thus $F = P$ in this case. The probability for a job arriving while the machine is busy is given by the utilization factor ρ. It follows that the probability of a job arriving in an idle period is $1 - \rho$.

It remains to find the flow-time of a job arriving during a busy period. As it is one particular job that is of interest, the previously used LCFS decomposition of the busy period can no longer be used. Denote instead the processing time of the initial job of a busy period by X_0. Let X_1 be the time needed to process all jobs that arrived during the processing of the initial job. Proceeding in this manner, X_j is the time needed to process all jobs that arrived in the previous interval of length X_{j-1}. Denote by N_j the number of jobs arriving during X_j. Finally, let $\psi_j(z)$ and $F_j(x)$ denote the Laplace transform and the probability distribution associated with X_j, respectively. An example of this FCFS decomposition of the busy period is shown in Figure 3 (Kleinrock, 1975, page 220).

It should be noted that Figure 2 and Figure 3 shows the same arrival pattern, but decomposed differently. This results in different sub-intervals. The length of the initial sub-interval however is the same in both figures.

By conditioning on N_{j-1} and X_{j-1} and by using that the processing

times are independent and identically distributed random variables, one can write

$$\begin{aligned}\mathrm{E}\left[e^{-zX_j} \mid X_{j-1}=x,\, N_{j-1}=n\right] &= \mathrm{E}\left[e^{-z\sum_{k=1}^{N_{j-1}} P_k}\right] \\ &= \mathrm{E}\left[e^{-zP_1}\right]\cdots\mathrm{E}\left[e^{-zP_n}\right] \\ &= [\gamma(z)]^n\end{aligned}$$

(Kleinrock, 1975, page 221). The last equality follows from the convolution property of the Laplace transform given by Equation (48) in Appendix A. Eliminating the conditions on N_{j-1} and X_{j-1} yields

$$\begin{aligned}\psi_j(z) &= \mathrm{E}\left[e^{-zX_j}\right] \\ &= \int_{x=0}^{\infty} e^{-\lambda x} \sum_{n=0}^{\infty} \frac{[\lambda x\gamma(z)]^n}{n!}\, \mathrm{d}F_{j-1}(x) \\ &= \int_{x=0}^{\infty} e^{-x[\lambda-\lambda\gamma(z)]}\, \mathrm{d}F_{j-1}(x) \\ &= \psi_{j-1}[\lambda-\lambda\gamma(z)]. \end{aligned} \tag{3}$$

Under the assumption that the system is non-saturated ($\rho < 1$) there will always be a finite j for which $X_j = 0$. A busy system will at some point become idle again. This can be written as

$$\lim_{j\to\infty} \psi_j(z) = 1. \tag{4}$$

Now, idle periods are completely ignored for a while and the focus is upon the busy periods. The system is defined to be in state j if the jth interval, X_j, of a busy period is in progress. Given that the system is busy, the probability of finding it in state j is

$$\Pr(\text{state } j \mid \text{busy}) = \frac{\mathrm{E}[X_j]}{\mathrm{E}[T]}, \tag{5}$$

where $\mathrm{E}[T]$ is the expected length of a busy period.

A job that arrives during X_j must wait until the current interval is finished and in addition it must wait for all jobs that arrived before it during X_j to be processed before it can go on the machine itself. Denote the remaining time of interval X_j when the job in question arrives as Y_j and let N be the number of jobs that arrived prior to this job during X_j. The flow-time of this job is then $F = Y_j + \sum_{i=1}^{N+1} P_i$.

The first step towards the Laplace transform of the flow-time distribution

is to condition on X_j, Y_j and N:

$$\begin{aligned}\mathrm{E}\left[e^{-zF} \mid X_j = x,\, Y_j = y,\, N = n\right] &= \mathrm{E}\left[e^{-z\left(y+\sum_{i=1}^{n+1} P_i\right)}\right] \\ &= \mathrm{E}\left[e^{-zy}\right] \mathrm{E}\left[e^{-zP_1}\right] \cdots \mathrm{E}\left[e^{-zP_{n+1}}\right] \\ &= e^{-zy}\left[\gamma(z)\right]^{n+1},\end{aligned}$$

where the last equality follows from the convolution property of the Laplace transform and the fact that y is fixed.

Removing the condition on N using that the probability of n arrivals in the time interval $X_j - Y_j$ is Poisson distributed with mean $\lambda(x-y)$ under the condition that $X_j = x$ and $Y_j = y$, yields

$$\begin{aligned}\mathrm{E}\left[e^{-zF} \mid X_j = x,\, Y_j = y\right] &= e^{-zy}[\gamma(z)]^{n+1} \sum_{n=0}^{\infty} e^{-\lambda(x-y)} \frac{[\lambda(x-y)]^n}{n!} \\ &= e^{-zy}\gamma(z)e^{-\lambda(x-y)} \sum_{n=0}^{\infty} \frac{[\lambda(x-y)\gamma(z)]^n}{n!} \\ &= \gamma(z)e^{-zy}e^{-\lambda(x-y)}e^{-\lambda(x-y)\gamma(z)}.\end{aligned}$$

Y_j is a random interception of the jth interval. From Equation (52) in Appendix B the joint density of Y_j and X_j is

$$\Pr\left(y < Y_j \leq y + \mathrm{d}y,\, x < X_j \leq x + \mathrm{d}x\right) = \frac{\mathrm{d}F_j(x)}{\mathrm{E}\left[X_j\right]}\mathrm{d}y$$

for $0 \leq y \leq x$ and $0 \leq x \leq \infty$. This can now be used to eliminate the condition on X_j and Y_j. The notation $\mathrm{E}\left[e^{-zF} \mid j\right]$ is used to denote the conditional Laplace transform associated with the flow-time of a job that arrives during the jth interval of a busy period.

$$\begin{aligned}\mathrm{E}\left[e^{-zF} \mid j\right] &= \gamma(z) \int_{x=0}^{\infty} \int_{y=0}^{x} e^{-x[\lambda-\lambda\gamma(z)]} e^{-y[z-\lambda+\lambda\gamma(z)]} \frac{\mathrm{d}F_j(x)}{\mathrm{E}\left[X_j\right]}\, \mathrm{d}y \\ &= \frac{\gamma(z)}{\mathrm{E}\left[X_j\right]\left[z-\lambda+\lambda\gamma(z)\right]} \\ &\quad \times \int_{x=0}^{\infty} \left[1 - e^{-x[z-\lambda+\lambda\gamma(z)]}\right] e^{-x[\lambda-\lambda\gamma(z)]}\, \mathrm{d}F_j(x) \\ &= \frac{\gamma(z)}{\mathrm{E}\left[X_j\right]\left[z-\lambda+\lambda\gamma(z)\right]} \\ &\quad \times \int_{x=0}^{\infty} \left[e^{-x[\lambda-\lambda\gamma(z)]} - e^{-xz}\right]\, \mathrm{d}F_j(x). \qquad (6)\end{aligned}$$

Replacing j by $j+1$ in Equation (3) yields

$$\psi_{j+1}(z) = \int_{x=0}^{\infty} e^{-x[\lambda-\lambda\gamma(z)]}\, \mathrm{d}F_j(x) = \psi_j\left[\lambda - \lambda\gamma(z)\right].$$

The first half of the integral in Equation (6) can be recognized as $\psi_{j+1}(z)$ and the second half as $\psi_j(z)$. Thus

$$\mathrm{E}\left[e^{-zF} \mid j\right] = \frac{\gamma(z)\left[\psi_{j+1}(z) - \psi_j(z)\right]}{\mathrm{E}\left[X_j\right]\left[z - \lambda + \lambda\gamma(z)\right]}.$$

By using Equation (5) the condition on state j can be removed and

$$\begin{aligned}\mathrm{E}\left[e^{-zF} \mid \text{busy}\right] &= \sum_{j=0}^{\infty} \frac{\mathrm{E}\left[X_j\right]}{\mathrm{E}\left[T\right]} \mathrm{E}\left[e^{-zF} \mid j\right] \\ &= \frac{\gamma(z)}{\mathrm{E}\left[T\right]\left[z - \lambda + \lambda\gamma(z)\right]} \sum_{j=0}^{\infty} \left[\psi_{j+1}(z) - \psi_j(z)\right]. \qquad (7)\end{aligned}$$

By Equation (4) the sum telescopes into $1 - \psi_1(z)$, which in this case simply is $1 - \gamma(z)$ since the first interval of the busy period consists of only one single processing time. Equation (2) gives the first moment of the busy period and the Laplace transform associated with flow-time of a job arriving during a busy period can be written

$$\mathrm{E}\left[e^{-zF} \mid \text{busy}\right] = \gamma(z) \frac{1 - \gamma(z)}{s\mathrm{E}\left[P\right]} \frac{z(1-\rho)}{z - \lambda + \lambda\gamma(z)}. \qquad (8)$$

The first part of this expression is simply the Laplace transform associated with processing time. The middle fraction is recognized from the renewal theory in Appendix B as the Laplace transform associated with residual life, in this situation the remaining time of the busy period in progress at time of arrival. The last term is the Laplace transform associated with the overall waiting time in a FCFS system (Kleinrock, 1975, page 200).

Since a job arrives at an idle system with probability $1 - \rho$ and finds the system busy with probability ρ, the unconditional Laplace transform associated with flow-time is

$$\begin{aligned}\phi_F(z) = \mathrm{E}\left[e^{-zF}\right] &= (1-\rho)\gamma(z) + \rho\mathrm{E}\left[e^{-zF} \mid \text{busy}\right] \\ &= (1-\rho)\gamma(z) + \rho\gamma(z) \frac{[1-\gamma(z)](1-\rho)}{\mathrm{E}\left[P\right]\left[z - \lambda + \lambda\gamma(z)\right]} \\ &= \frac{z(1-\rho)}{z - \lambda + \lambda\gamma(z)}\gamma(z).\end{aligned}$$

The last simplification is simply obtained by using $\rho = \lambda\mathrm{E}\left[P\right]$.

The Laplace transform associated with waiting time W is $\phi_W(z)$. Recall that $F = W + P$. This implies that $\phi_F(z) = \phi_W(z)\gamma(z)$ and from $\phi_F(z)$ above

$$\phi_W(z) = \frac{z(1-\rho)}{z - \lambda + \lambda\gamma(z)}$$

is obtained directly.

By evaluating the limit of the derivative of $\phi_W(z)$ as z goes to zero the first moment of waiting time can, by application of l'Hôpital's rule, be obtained as

$$\mathrm{E}[W] = \frac{\lambda \mathrm{E}\left[P^2\right]}{2(1-\rho)}.$$

This is known as the Pollaczek-Khinchine formula for expected waiting time in a FCFS queuing system (Ross, 2007, page 529).

For flow-time the first moment is

$$\mathrm{E}[F] = \mathrm{E}[P] + \mathrm{E}[W] = \mathrm{E}[P] + \frac{\lambda \mathrm{E}\left[P^2\right]}{2(1-\rho)}.$$

4 Delay cycles

A delay cycle is a generalization of the the busy periods discussed in Section 3. Analysis of delay cycles provides a powerful method for obtaining results for queuing systems with selection disciplines based on priority rules. Different priority rules will be discussed in great details in Section 5, Section 6 and Section 7. This section deals with delay cycles as a concept. Similar derivations of the properties of interest are given by Conway et al. (1967, Chapter 8-3), Kleinrock (1976, Chapter 3.3) and Takagi (1991, Chapter 1.2).

4.1 Duration of a delay cycle

A delay cycle begins with an initial delay of length T_0 that is the performance of some task other than the processing of an initially arriving job. During the initial delay no ordinary jobs are being processed by the machine.

The initial delay may be shorter or longer than the processing time of a ordinary job, causing the delay cycle to be more general than a busy period. Allowing the initial delay to be the processing time of an initially arriving, ordinary job reduces the delay cycle to a busy period.

Typically the initial delay is the remaining processing time of some special task when an ordinary jobs arrives an otherwise empty system. It follows that the system is not necessarily idle prior to the initiation of the delay cycle. Several ordinary jobs may have arrived during the initial delay. When the initial delay ends, each of these jobs will generate its own sub-busy period. Together these sub-busy periods form the delay busy period T_b.

The delay cycle ends when there are no more ordinary jobs left in the system. The duration of the entire delay cycle is denoted T_c and the relationship $T_c = T_0 + T_b$ is obvious.

T_0, T_b, and T_c are random variables distributed according to $H_0(t)$, $H_b(t)$ and $H_c(t)$, respectively. The Laplace transforms of the corresponding probability density functions are denoted $\eta_0(z)$, $\eta_b(z)$ and $\eta_c(z)$. In the following

$\eta_0(z)$ will be assumed known. Usually this assumption do not cause any problems as $\eta_0(z)$ very often, as pointed out by both Conway et al. (1967, page 151) and Kleinrock (1976, page 112), is in fact known or at least easily calculated.

Assume that N ordinary jobs arrive at the system during the initial delay of length T_0. By conditioning on T_0 and N the Laplace transform associated with the delay busy period, $\eta_b(z) = \mathrm{E}[e^{-zT_b}]$, can be calculated. By the convolution property of the Laplace transform, Equation (48) in Appendix A,

$$\mathrm{E}\left[e^{-zT_b} \mid T_0 = t,\, N = n\right] = [\eta(z)]^n\,,$$

since the jobs arriving during T_0 generates n sub-busy periods, which are independent and distributed exactly as a sub-busy period (Kleinrock, 1976, page 112). As in Section 3, T denotes the duration of a busy period and $\eta(z)$ is its corresponding Laplace transform.

N is Poisson distributed with mean λt under the condition that $T_0 = t$. Removing the condition on N yields

$$\mathrm{E}\left[e^{-zT_b} \mid T_0 = t\right] = \sum_{n=0}^{\infty} [\eta(z)]^n \frac{(\lambda t)^n}{n!} e^{-\lambda t} = e^{-t[\lambda - \lambda\eta(z)]}.$$

The next step is to also avoid the condition on T_0. This is done simply by integrating over $H_0(t)$. Thus

$$\mathrm{E}\left[e^{-zT_b}\right] = \int_{t=0}^{\infty} e^{-t[\lambda - \lambda\eta(z)]}\, \mathrm{d}H_0(t)$$

and, by observing that the integral is simply the Laplace transform of the probability density function of T_0 evaluated at $\lambda - \lambda\eta(z)$, the final result is obtained as

$$\eta_b(z) = \eta_0[\lambda - \lambda\eta(z)]. \tag{9}$$

$\eta_c(z)$ is obtained in a similar fashion using the fact that $T_c = T_0 + T_b$. Again by conditioning on T_0 and N

$$\begin{aligned}\mathrm{E}\left[e^{-zT_c} \mid T_0 = t,\, N = n\right] &= e^{-zt}\mathrm{E}\left[e^{-zT_b} \mid T_0 = t,\, N = n\right] \\ &= e^{-zt}[\eta(z)]^n.\end{aligned}$$

Avoiding the conditions on T_0 and N exactly as above yields

$$\begin{aligned}\eta_c(z) = \mathrm{E}\left[e^{-zT_c}\right] &= \int_{t=0}^{\infty} e^{-zt} \sum_{n=0}^{\infty} [\eta(z)]^n \frac{(\lambda t)^n}{n!} e^{-\lambda t}\, \mathrm{d}H_0(t) \\ &= \eta_0[z + \lambda - \lambda\eta(z)]\end{aligned} \tag{10}$$

Moments of T_b and T_c can be found by the same procedure as earlier. That is, derivation and limit evaluation of the Laplace transforms as z goes

to zero. This procedure yields, for T_b

$$\mathrm{E}\left[T_b\right] = \frac{\rho \mathrm{E}\left[T_0\right]}{1-\rho}$$

and

$$\mathrm{E}\left[T_b^2\right] = \frac{\lambda \mathrm{E}\left[P^2\right]}{(1-\rho)^3}\mathrm{E}\left[T_0\right] + \frac{\rho^2}{(1-\rho)^2}\mathrm{E}\left[T_0^2\right]$$

(Conway et al., 1967, page 151). In very similar manners the first and second moment of T_c can be found to be

$$\mathrm{E}\left[T_c\right] = \frac{\mathrm{E}\left[T_0\right]}{1-\rho} \tag{11}$$

and

$$\mathrm{E}\left[T_c^2\right] = \frac{\lambda \mathrm{E}\left[P^2\right]}{(1-\rho)^3}\mathrm{E}\left[T_0\right] + \frac{\mathrm{E}\left[T_0^2\right]}{(1-\rho)^2}. \tag{12}$$

Takagi (1991, page 28) states the third moment as well, but it is of little interest here.

4.2 Flow-time in a delay cycle

The next and final step is to generalize the Laplace transform of flow-time of a job arriving during a busy period, given by Equation (8) in Section 3, to obtain the Laplace transform associated with the flow-time of a job arriving during a delay cycle.

All jobs processed in a delay cycle arrive when the machine is busy. The expected duration of the busy period T is thus replaced with the expected duration of the delay cycle T_c. Furthermore, the jobs generating the sub-busy periods arrive during the initial delay T_0 and not during a processing interval P. Hence, the telescoping sum in Equation (7) reduces to $1 - \eta_0(z)$ in the generalized case.

Since the focus is upon jobs arriving during a delay cycle, the analogue to $\mathrm{E}\left[e^{-zF} \mid \text{busy}\right]$ in Equation (8) is directly the Laplace transform associated with flow-time in a delay cycle. This is denoted $\phi_{F|c}(z)$. It is obtained by the above discussed modification of Equation (7) as

$$\phi_{F|c}(z) = \frac{(1-\rho)\gamma(z)[1-\eta_0(z)]}{\mathrm{E}\left[T_0\right]\left[z-\lambda+\lambda\gamma(z)\right]} \tag{13}$$

(Conway et al., 1967, page 155), since $\mathrm{E}\left[T_c\right] = \mathrm{E}\left[T_0\right]/(1-\rho)$ from Equation (11).

As a closing remark to this section, waiting time is still flow-time minus processing time. The Laplace transform associated with waiting time in a delay cycle is thus

$$\phi_{W|c}(z) = \frac{(1-\rho)[1-\eta_0(z)]}{\mathrm{E}\left[T_0\right]\left[z-\lambda+\lambda\gamma(z)\right]}. \tag{14}$$

5 Non-preemptive priority systems

The standard $M/G/1$ queuing system discussed earlier will now be modified in such a way that the selection discipline is no longer FCFS, but rather a non-preemptive priority discipline.

In the first part of this section a general description of the non-preemptive priority discipline is provided. Some of its properties are then explored in more detail. Finally flow-times will be studied through an analysis of delay cycles.

Non-preemptive priority systems are treated by Conway et al. (1967, Chapter 8-6), Kleinrock (1976, Chapter 3.6) and Takagi (1991, Chapter 3.2).

5.1 Properties of a non-preemptive priority system

The arriving jobs are divided into K different priority classes and each class is assigned a level of priority. It is a convention in the queuing literature to assign the highest priority level to the class 1 jobs. That is, jobs from class k has priority above all class $k+1,\ldots,K$ jobs.

The selection discipline is such that of the jobs waiting in queue, the one with highest priority is selected for processing next. Within each priority class the discipline is still FCFS. If a high priority job arrives while a job of lower priority is being processed, the later arriving job will have to wait, despite its higher level of priority. The lower prioritized job is allowed to complete its processing and thus, preemptions never occur. Therefore this selection discipline is said to be non-preemptive priority.

In general the different priority classes can have different arrival rates and processing distributions. Subscript k denotes class k. λ_k and P_k denotes the arrival rate and processing time of a class k job, respectively. P_k is distributed according to $G_k(p)$, and the Laplace transform associated with the probability density function of P_k is $\gamma_k(z)$. The utilization factor for class k is $\rho_k = \lambda_k \mathrm{E}[P_k]$; the proportion of time the machine is busy processing class k jobs. The overall utilization factor for the entire system is

$$\rho = \sum_{j=1}^{K} \lambda_j \mathrm{E}[P_j] .$$

In a similar fashion flow-time and waiting time for a given class k will be denoted F_k and W_k, respectively.

The focus in the following is upon jobs belonging to a particular class k. If it were not for the other $K-1$ classes the system would be a standard FCFS system with only class k jobs. Jobs from the other priority classes may be viewed as disturbing elements, disrupting the system seen from a class k job's point of view in two ways: First, while a class k job is being processed arrivals of higher priority jobs may cause the selection of the next

class k job to be delayed. Second, an idle period may terminate with the arrival of some job from another class, causing the ensuing busy period to be a delay cycle rather than a simple busy period for class k.

It is convenient to group the other $K-1$ classes into jobs with priority above the class k jobs (class a) and jobs with priority below the class k jobs (class b). For class a the total arrival rate is

$$\Lambda_a = \sum_{j=1}^{k-1} \lambda_j,$$

since class a consists of all classes from 1 to $k-1$. Similarly class b consists of all classes from $k+1$ to K. Thus the class b arrival rate is given as

$$\Lambda_b = \sum_{j=k+1}^{K} \lambda_j.$$

P_a and P_b are the random variables denoting processing time of the jobs in each group, distributed according to

$$G_a(p) = \frac{1}{\Lambda_a} \sum_{j=1}^{k-1} \lambda_j G_j(p)$$

and

$$G_b(p) = \frac{1}{\Lambda_b} \sum_{j=k+1}^{K} \lambda_j G_j(p),$$

respectively. The corresponding Laplace transforms are $\gamma_a(s)$ and $\gamma_b(s)$. Finally the utilization factors for class a and class b are $\rho_a = \Lambda_a \mathrm{E}[P_a]$ and $\rho_b = \Lambda_b \mathrm{E}[P_b]$, respectively. This notation is according to Conway et al. (1967, page 160).

5.2 Blocking time

To deal with the first deviation from a FCFS system with only class k jobs it is convenient to define blocking time[2]. While a class k job is being processed arrivals of higher priority jobs may cause the selection of the next class k job to be delayed. Blocking time account for this disruption by taking the role of class k processing time. Denoted B_k, blocking time is the processing time of a class k job plus the processing time of all higher priority jobs arriving during this processing time. It is thus really a delay cycle, the initial delay being a class k processing time and all jobs processed in the delay busy period are from class a.

[2]This is not the same as blocking time in the railway theory

Denoting the Laplace transform associated with the distribution of blocking time by $\phi_{B_k}(z)$, from Equation (10) it follows that

$$\phi_{B_k}(z) = \gamma_k[z + \Lambda_a - \Lambda_a \eta_a(z)], \tag{15}$$

where $\eta_a(z) = \gamma_a[z + \Lambda_a - \Lambda_a \eta_a(z)]$ is the Laplace transform associated with the distribution of a class a busy period (see Equation (1)). From this, the first moment of the blocking time can be found as

$$\begin{aligned} \mathrm{E}[B_k] &= \lim_{z\to 0}(-1)\frac{\mathrm{d}\phi_{B_k}(z)}{\mathrm{d}z} \\ &= \lim_{z\to 0}(-1)\gamma_k'[z + \Lambda_a - \Lambda_a\eta_a(z)]\frac{\mathrm{d}}{\mathrm{d}z}[z + \Lambda_a - \Lambda_a\eta_a(z)] \\ &= \mathrm{E}[P_k]\,[1 - \Lambda_a\eta_a'(0)], \end{aligned}$$

since $\eta_a(0) = 1$. Furthermore, $\eta_a'(0) = \gamma_a'(0)[1 - \Lambda_a\eta_a'(0)]$ and thus

$$\begin{aligned} \mathrm{E}[B_k] &= \mathrm{E}[P_k]\left[1 - \frac{\Lambda_a\gamma_a'(0)}{1 + \Lambda_a\gamma_a'(0)}\right] \\ &= \frac{\mathrm{E}[P_k]}{1 - \rho_a}, \end{aligned} \tag{16}$$

since $\Lambda_a\gamma_a'(0) = -\rho_a$. Blocking time is the effective processing time of a class k job observed by other class k jobs waiting for processing.

It is convenient to define

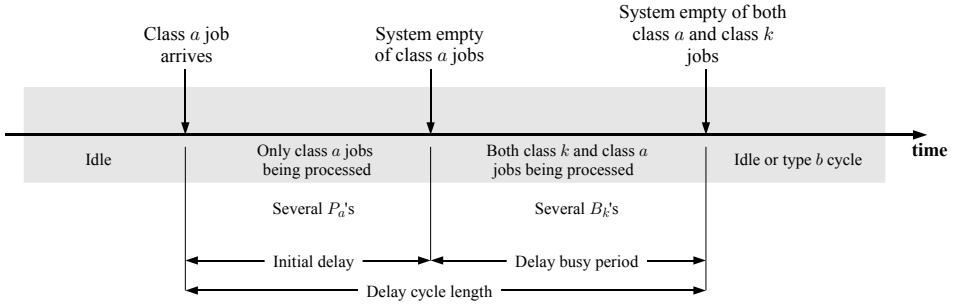
$$\rho_{B_k} = \lambda_k \mathrm{E}[B_k] = \frac{\rho_k}{1 - \rho_a}.$$

That is, ρ_{B_k} is the expected number of class k jobs arriving at the system during a blocking time. This quantity will play the role ρ played in the standard $M/G/1$ FCFS system. This since the processing of a class b job has no effect on the class k jobs beyond that of being a possible initial delay in a delay cycle. For a class k job waiting, the machine is occupied after the initial delay only if other k jobs or a jobs are being processed before itself.

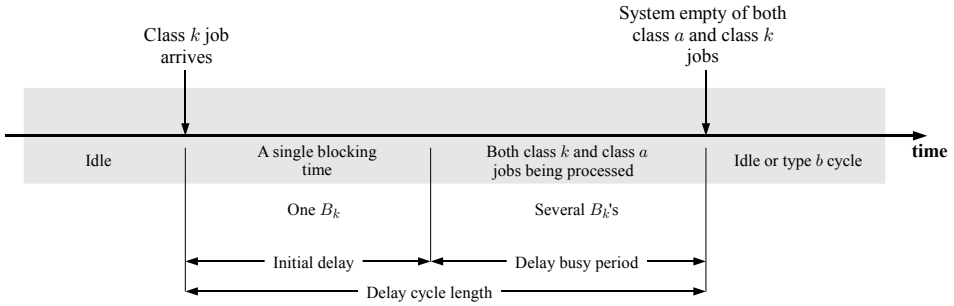
5.3 Three types of delay cycles

As mentioned, an idle period may terminate with the arrival of some job from another class, causing the ensuing busy period to be a delay cycle rather than a simple busy period for class k. This can be handled by considering the state of the system at the time of a class k arrival.

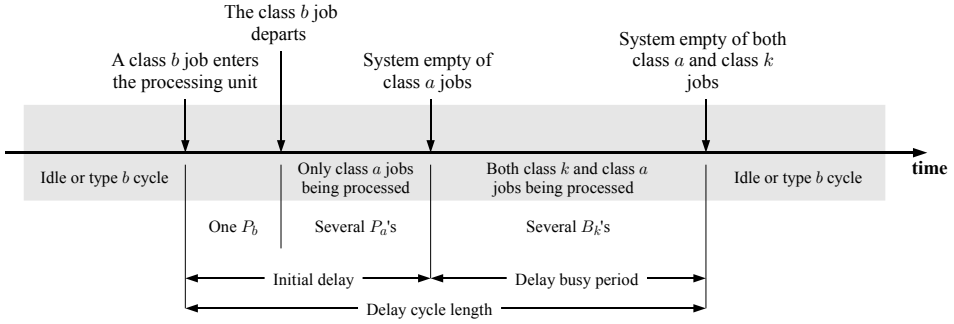
Some class k jobs arrive when the machine is idle and experiences no waiting time at all. All other class k jobs arrive during delay cycles, of which there are three different types. These are called type a, type k and type b cycles depending on the priority class of the job initiating them. All three types of cycles are shown schematically in Figure 4. This figure is



(a) Type a delay cycle



(b) Type k delay cycle



(c) Type b delay cycle

Figure 4: The different types of delay cycles in a non-preemptive priority system.

inspired from figures given by Conway et al. (1967, page 162). Any of these delay cycles ends when a process completion leaves the system empty of both class a and class k jobs.

From Equation (13) the Laplace transform associated with flow-time in the various types of delay cycles can be obtained by suitable modification.

As mentioned, in all three delay cycles of interest ρ in Equation (13) is replaced by ρ_{B_k}. For similar reasons $\gamma(z)$ is replaced by $\phi_{B_k}(z)$ since it is no longer only the processing time of a class k job that is of interest, but in addition the processing time of those higher priority jobs arriving during the processing time of that class k job.

It is clear that $\mathrm{E}[T_c] = \mathrm{E}[T_0]/(1-\rho)$ in Equation (13) must be replaced by $\mathrm{E}[T_{cj}] = \mathrm{E}[T_{0j}]/(1-\rho_{B_k})$, where T_{0j} denotes the initial delay in a type j delay cycle. As a natural extension of this notation $\eta_{0j}(z)$ denotes the Laplace transform associated with initial delay in a type j cycle.

These modifications result in a Laplace transform associated with the flow-time in a type j cycle given as

$$\phi_{F_k|j}(z) = \frac{(1-\rho_{B_k})\gamma_k(z)[1-\eta_{0j}(z)]}{\mathrm{E}[T_{0j}]\,[z-\lambda_k+\lambda_k\phi_{B_k}(z)]}. \tag{17}$$

In a type a delay cycle the initial delay, denoted T_{0a}, is a busy period involving class a jobs only. That is, the Laplace transform associated with T_{0a} is simply $\eta_{0a}(z) = \eta_a(z)$. From Equation (2)

$$\mathrm{E}[T_{0a}] = \frac{\mathrm{E}[P_a]}{1-\rho_a}.$$

Inserting the above into Equation (17) and observing that $(1-\rho_a)(1-\rho_{B_k}) = 1-\rho_a-\rho_k$ yields

$$\phi_{F_k|a}(z) = \frac{(1-\rho_a-\rho_k)\gamma_k(z)[1-\eta_a(z)]}{\mathrm{E}[P_a]\,[z-\lambda_k+\lambda_k\phi_{B_k}(z)]}.$$

for a type a cycle.

For a type k cycle the initial delay is a blocking time. It follows that $T_{0k} = B_k$ and $\eta_{0k}(z) = \phi_{B_k}(z)$. The first moment of blocking time is given in Equation (16). Hence

$$\phi_{F_k|k}(z) = \frac{(1-\rho_a-\rho_k)\gamma_k(z)[1-\phi_{B_k}(z)]}{\mathrm{E}[P_k]\,[z-\lambda_k+\lambda_k\phi_{B_k}(z)]}.$$

The expression for type b cycles becomes somewhat more complex. The initial delay is itself a delay cycle where the initial delay is the processing time of a class b job, P_b. The jobs in the delay busy period are all from class a, since the class k job of interest is the first in line. Hence, the Laplace transform associated with initial delay in a type b cycle is

$$\eta_{0b}(z) = \gamma_b[z+\Lambda_a-\Lambda_a\eta_a(z)].$$

From this the first moment of T_{0b} can be found as

$$\mathrm{E}[T_{0b}] = \frac{\mathrm{E}[P_b]}{1-\rho_a}.$$

Finally, the Laplace transformation associated with waiting time in a type b delay cycle can be obtained as

$$\phi_{F_k|b}(z) = \frac{(1-\rho_a-\rho_k)\gamma_k(z)(1-\gamma_b[z+\Lambda_a-\Lambda_a\eta_a(z)])}{\mathrm{E}\left[P_b\right]\left[z-\lambda_k+\lambda_k\phi_{B_k}(z)\right]}.$$

By conditioning on the proportion of time the system is engaged in the different types of delay cycles, an expression for the overall expected flow-time in a non-preemptive priority queue can be found. Before that expression can be obtained, the probabilities of finding the system engaged in one of the three types of delay cycles or in an idle state, must be calculated.

5.4 Expected flow-time

The system is defined to be in state 0 when the machine is idle. When the machine is engaged in either an a, k or b cycle, the system is in state a, k or b, respectively. π_j is the steady state probability of state j, or the proportion of time the system is in state $j = \{0, a, k, b\}$ in the long run. The following calculation of these steady state probabilities mainly follows Conway et al. (1967, page 163–164), but is somewhat simplified by considering an observation made by Drekic and Stanford (2000, page 295). By defining m_j as the mean length of a j cycle and l_j as the mean time between the system enters a type j cycle, π_j can be expressed as

$$\pi_j = \lim_{t\to\infty} \Pr\left(\text{state } j \text{ at time } t\right) = \frac{m_j}{l_j}, \tag{18}$$

where $m_j = \mathrm{E}\left[T_{cj}\right]$.

$\rho = \rho_a + \rho_k + \rho_b$ is the system utilization as a non-idle system will be engaged in either an a, k or b cycle, and clearly

$$\pi_0 = 1 - \rho.$$

Let $N_j(t)$ be the number of busy periods in the interval of length t initiated by an arrival of a class j job at the system in idle state. r_j is the rate at which such busy periods occur. By the elementary renewal theorem given in Appendix B, Equation (55),

$$r_j = \lim_{t\to\infty} \frac{\mathrm{E}\left[N_j(t)\right]}{t},$$

and hence, the mean time between entrances to state j is $l_j = 1/r_j$.

Since type a cycles only can be initiated by an arrival of a class a job at an idle system, it is clear that type a cycles occur by a rate $r_a = \Lambda_a\pi_0$ (Drekic and Stanford, 2000, page 295) and

$$l_a = \frac{1}{\Lambda_a(1-\rho)}.$$

The same argument holds for a type k cycle. $r_k = \lambda_k \pi_0$ and thus

$$l_k = \frac{1}{\lambda_k(1-\rho)}.$$

While type a and k cycles are initiated by class a and k jobs arriving at an idle system only, every class b job is responsible for initiating a type b cycle. That is, each type b cycle consists of only one class b job and there is on average Λ_b type b cycles per unit time. Again from the elementary renewal theorem, Equation (55), one find

$$l_b = 1/\Lambda_b.$$

For $j = \{a, k, b\}$ the m_j's are found as

$$m_j = \frac{\mathrm{E}\left[T_{0j}\right]}{1-\rho_{B_k}}$$

where $\rho_{B_k} = \lambda_k \mathrm{E}\left[P_{B_k}\right]$ from the first moment of a delay cycle given in Equation (11). $\mathrm{E}\left[T_{0j}\right]$ is known from the above discussion of the different types of delay cycles and thus

$$m_a = \frac{\mathrm{E}\left[P_a\right]}{1-\rho_a-\rho_k},$$

$$m_k = \frac{\mathrm{E}\left[P_k\right]}{1-\rho_a-\rho_k}$$

and

$$m_b = \frac{\mathrm{E}\left[P_b\right]}{1-\rho_a-\rho_k}.$$

All quantities needed for calculation of the π_j's are now known. $\pi_0 = 1-\rho$ as discussed. π_a, π_k and π_b are given as

$$\pi_a = \frac{m_a}{l_a} = \frac{\rho_a(1-\rho)}{1-\rho_a-\rho_k},$$

$$\pi_k = \frac{m_k}{l_k} = \frac{\rho_k(1-\rho)}{1-\rho_a-\rho_k},$$

and

$$\pi_b = \frac{m_b}{l_b} = \frac{\rho_b}{1-\rho_a-\rho_k}.$$

The unconditional Laplace transform $\phi_{F_k}(z)$ associated with the distribution of flow-time of a class k job under the non-preemptive priority

discipline can now be obtained. Note that the flow-time for a job arriving at an idle system is simply P_k and that $\phi_{F_k|0}(z) = \gamma_k(z)$.

$$\begin{aligned}\phi_{F_k}(z) &= \mathrm{E}\left[e^{-zF_k}\right] \\ &= \pi_0\gamma_k(z) + \pi_a\phi_{F_k|a}(z) + \pi_k\phi_{F_k|k}(z) + \pi_b\phi_{F_k|b}(z) \\ &= \gamma_k(z)\frac{(1-\rho)[z+\Lambda_a-\Lambda_a\eta_a(z)] + \lambda_b(1-\gamma_b[z+\Lambda_a-\Lambda_a\eta_a(z)])}{\lambda_k\phi_{B_k}(z) - \lambda_k + z} \\ &= \gamma_k(z)\frac{(1-\rho)[z+\Lambda_a-\Lambda_a\eta_a(z)] + \lambda_b(1-\gamma_b[z+\Lambda_a-\Lambda_a\eta_a(z)])}{\lambda_k\gamma_k[z+\Lambda_a-\Lambda_a\eta_a(z)] - \lambda_k + z},\end{aligned}$$

where the last equality is by Equation (15).

It is a tedious, but straight forward procedure to obtain the first moment of flow-time for a class k job in a non-preemptive priority queue. The limit of the derivative of $\phi_{F_k}(z)$ must be evaluated as z goes to zero. To obtain the result for $-\phi'_{F_k}(0)$ it is necessary to apply l'Hôpital's rule twice. When done, some algebraic manipulation yields

$$\mathrm{E}[F_k] = \mathrm{E}[P_k] + \frac{\Lambda_a\mathrm{E}\left[P_a^2\right] + \lambda_k\mathrm{E}\left[P_k^2\right] + \Lambda_b\mathrm{E}\left[P_b^2\right]}{2(1-\rho_a-\rho_b)(1-\rho_a)}. \tag{19}$$

6 Preemptive priority systems

Of interest now is a priority discipline which the literature refers to as preemptive. It is characterized by the following property: Jobs of higher priority are allowed to preempt jobs of lower priority and immediately go on the machine for processing at the time of arrival. The job of lower priority returns to the head of the queue of its class. When the system is empty of all higher priority jobs, the preempted job goes on the machine for another processing attempt.

There are three different basic variations of the preemptive priority discipline discussed in the literature. The somewhat simpler version is that of preemptive resume, where the processing is continued from where it left off when the job was preempted. Under this selection discipline no processing time is wasted and the system is work conserving. This is not the case for the two preemptive repeat disciplines.

Under the preemptive repeat disciplines the processing starts over when a job reenters the processing unit after a preemption. When the processing time needed for completion is drawn only once – the first time the job goes on the machine – the selection discipline is called preemptive repeat-equal.

While Conway et al. (1967) treat all the three cases in parallel, here it is only the third case, preemptive repeat-different, that is of interest. In this case the processing time needed for completion of a given job is drawn from the processing time distribution every time the job enters the processing unit.

This section is based closely on Conway et al. (1967, Chapter 8-7). Preemptive priority queues are also discussed by Takagi (1991, Chapter 3.4).

6.1 The preemptive repeat-different priority discipline

The analysis of the preemptive repeat-different system is similar to the analysis of the non-preemptive priority system. Still a given class k job is considered, and the focus is on how the preemptive repeat-different system differs from the FCFS system with only class k jobs. The notation must be extended somewhat as there is a need to consider several random variables not introduced earlier.

First, W_k is still the waiting time for a class k job. It is defined as the time interval between the moment a class k job arrives at the system and the moment when the first processing attempt begins.

The rest of the time a class k job spends in the system is called residence time, denoted by R_k. In other words, it is the time interval between the first entrance to the processing facility and process completion. Residence time plays the role in a preemptive repeat-different priority system that blocking time played in the non-preemptive priority system and processing time played in the standard FCFS system. R_k is the effective processing time of a class k job observed by another class k job waiting for processing.

Flow-time for a class k job F_k is the entire time between arrival at the system and process completion, that is

$$F_k = W_k + R_k.$$

An important random variable in the following is the duration of a preemption. For a class k job this is denoted by D_k and defined as the time from when a preemption first occur to the next processing attempt. It is common to call this variable the breakdown time of a class k job. Properties of the breakdown time will be discussed in Section 6.5.

Under the preemptive repeat-different discipline processing time spent on a job that is preempted goes to waste. The wasted processing time of a class k job is the total length of all the non-completed service attempts. The final processing attempt, in which the processing is completed, is called successful processing time. Wasted and successful processing time for a class k job are denoted P_{wk} and P_{sk}, respectively. Their sum, $P_{gk} = P_{wk} + P_{sk}$, is called gross processing time.

It is now obvious that the preemptive repeat-different priority system is not work conserving. The average amount of time a class k job keeps the processing unit busy is $\mathrm{E}[P_{gk}]$. Thus, the class k utilization factor is now

$$\tilde{\rho}_k = \lambda_k \mathrm{E}[P_{gk}].$$

Because of a recursive relationship between residence time and breakdown time which will be explored towards the end of this section, it is

convenient to introduce the notation

$$\Lambda_k = \sum_{j=1}^{k} \lambda_j.$$

When a class k job suffers a preemption, only one higher priority job can be present in the system, namely the one job that caused the preemption. Class a still refers to the jobs of priority above that of class k and class b to the jobs of priority below. Now, for a class k job in process a preemption will occur with intensity Λ_{k-1}, earlier represented by Λ_a.

In the following N denotes the number of preemptions a job suffers before it is completed.

6.2 Wasted and successful processing time

Let P_k be the processing time needed to complete a given processing attempt of a class k job. Let Y denote the interval from the start of the processing attempt of the class k job to the arrival of a class a job. As a preemption occurs with intensity Λ_{k-1}, Y is exponentially distributed with parameter Λ_{k-1}. That is, $F_Y(y) = 1 - e^{-\Lambda_{k-1}y}$ and $f_Y(y) = \Lambda_{k-1}e^{-\Lambda_{k-1}y}$. P_k is, as always, distributed according to $G_k(p)$. The joint density of Y and P_k is

$$\Pr\left(y < Y \leq y + \mathrm{d}y,\, p < P_k \leq p + \mathrm{d}p\right) = \Lambda_{k-1}e^{-\Lambda_{k-1}y}\mathrm{d}y\mathrm{d}G_k(p). \qquad (20)$$

The probability that a given attempt of processing a class k job is successful is the probability of Y being greater than, or equal to, P_k. It is a straightforward procedure to obtain $\Pr(Y \geq P_k)$ from the joint density above:

$$\begin{aligned}\Pr\left(Y \geq P_k\right) &= \int_{p=0}^{\infty}\int_{y=p}^{\infty} \Lambda_{k-1}e^{-\Lambda_{k-1}y}\ \mathrm{d}y\ \mathrm{d}G_k(p) \\ &= \int_{p=0}^{\infty} e^{-\Lambda_{k-1}p}\ \mathrm{d}G_k(p) \\ &= \gamma_k(\Lambda_{k-1}).\end{aligned}$$

The probability of a processing attempt to succeed is thus $\gamma_k(\Lambda_{k-1})$. As the corresponding probability of suffering a preemption is $1-\gamma_k(\Lambda_{k-1})$ it is clear that the number of preemptions N suffered by a class k job is geometrically distributed with parameter $\gamma_k(\Lambda_{k-1})$. That is

$$\Pr\left(N = n\right) = [1 - \gamma_k(\Lambda_{k-1})]^n\gamma_k(\Lambda_{k-1}).$$

The length of a wasted processing interval is Y under the condition that

$Y < P_k$. Hence

$$\begin{aligned} g_{wk}(y)\mathrm{d}y &= \Pr\left(y < P_{wk} \leq y + \mathrm{d}y\right) \\ &= \frac{\Pr\left(y < Y \leq y + \mathrm{d}y,\, Y < P_k\right)}{\Pr\left(Y < P_k\right)} \\ &= \frac{1}{1 - \gamma_k(\Lambda_{k-1})} \int_{p=y}^{\infty} \Lambda_{k-1} e^{-\Lambda_{k-1} y}\ \mathrm{d}G_k(p)\ \mathrm{d}y. \end{aligned}$$

It is now possible to obtain the Laplace transform $\gamma_{wk}(z)$ associated with the wasted processing time P_{wk} as

$$\begin{aligned} \gamma_{wk}(z) &= \int_{y=0}^{\infty} e^{-zy} g_{wk}(y)\ \mathrm{d}y \\ &= \int_{y=0}^{\infty} e^{-zy} \frac{1}{1 - \gamma_k(\Lambda_{k-1})} \int_{p=y}^{\infty} \Lambda_{k-1} e^{-\Lambda_{k-1} y}\ \mathrm{d}G_k(p)\ \mathrm{d}y \\ &= \frac{\Lambda_{k-1}}{1 - \gamma_k(\Lambda_{k-1})} \int_{y=0}^{\infty} \int_{p=y}^{\infty} e^{-(z+\Lambda_{k-1})y}\ \mathrm{d}G_k(p)\ \mathrm{d}y \\ &= \frac{\Lambda_{k-1}}{1 - \gamma_k(\Lambda_{k-1})} \int_{y=0}^{\infty} e^{-(z+\Lambda_{k-1})y} [1 - G_k(y)]\ \mathrm{d}y \qquad (*) \\ &= \frac{\Lambda_{k-1}}{1 - \gamma_k(\Lambda_{k-1})} \left[\frac{(1 - \gamma_k(z + \Lambda_{k-1})}{z + \Lambda_{k-1}}\right] \qquad (**) \\ &= \frac{\Lambda_{k-1}}{z + \Lambda_{k-1}} \frac{1 - \gamma_k(z + \Lambda_{k-1})}{1 - \gamma_k(\Lambda_{k-1})}. \qquad (21) \end{aligned}$$

Here $(*)$ follows by the same arguments that lead to Equation (53) in Appendix B and $(**)$ follows from Equation (46) and Equation (47) in Appendix A, analogous to the result obtained in Equation (54).

The development of the Laplace transform associated with successful processing time P_{sk} follows along the same path, but the calculations are in fact simpler. A successful processing interval is simply P_k under the condition that $Y \geq P_k$. Using Equation (20) it is easy to obtain the relative occurrence of a successful processing time p as

$$\begin{aligned} g_{sk}(p)\mathrm{d}p &= \Pr\left(p < P_{sk} \leq p + \mathrm{d}p\right) \\ &= \frac{\Pr\left(p < P_k \leq p + \mathrm{d}p,\, Y \geq P_k\right)}{\Pr\left(Y \geq P_k\right)} \\ &= \frac{1}{\gamma_k(\Lambda_{k-1})} \int_{y=p}^{\infty} \Lambda_{k-1} e^{-\Lambda_{k-1} y}\ \mathrm{d}y\ \mathrm{d}G_k(p) \\ &= \frac{1}{\gamma_k(\Lambda_{k-1})} e^{-\Lambda_{k-1} p}\ \mathrm{d}G_k(p). \end{aligned}$$

Finally, the Laplace transform associated with successful processing time of

a class k job can be found to be

$$\begin{aligned}\gamma_{sk}(z) &= \int_{p=0}^{\infty} e^{-zp} g_{sk}(p)\mathrm{d}p \\ &= \frac{1}{\gamma_k(\Lambda_{k-1})} \int_{p=0}^{\infty} e^{-p(z+\Lambda_{k-1})} \mathrm{d}G_k(p) \\ &= \frac{\gamma_k(z+\Lambda_{k-1})}{\gamma_k(\Lambda_{k-1})}. \end{aligned} \tag{22}$$

6.3 Residence time and gross processing time

For a class k job each preemption is followed by a breakdown time. The residence time for a class k job under a preemptive repeat-different rule therefore consists of N realizations of $P_{wk} + D_k$ followed by one realization of P_{sk}.

Given that a job is preempted n times, the residence time R_k is the sum of $2n+1$ random variables. The n wasted processing times, the n breakdown times and the one successful processing interval are all independent of each other. The conditional Laplace transform associated with residence time can thus be written

$$\begin{aligned}\mathrm{E}\left[e^{-zR_k} \mid N = n\right] &= \mathrm{E}\left[e^{-zP_{sk}}\right] \left(\mathrm{E}\left[e^{-zP_{wk}}\right] \mathrm{E}\left[e^{-zD_k}\right]\right)^n \\ &= \gamma_{sk}(z)[\gamma_{wk}(z)\phi_{D_k}(z)]^n,\end{aligned}$$

where $\phi_{D_k}(z)$ is the Laplace transform associated with breakdown time.

The Laplace transform associated with residence time can be obtained by removing the condition on N, which is geometrically distributed with parameter $\gamma_k(\Lambda_{k-1})$. Thus

$$\begin{aligned}\phi_{R_k}(z) &= \mathrm{E}\left[e^{-zR_k}\right] \\ &= \sum_{n=0}^{\infty} \mathrm{E}\left[e^{-zR_k} \mid N = n\right] \Pr\left(N = n\right) \\ &= \sum_{n=0}^{\infty} \gamma_{sk}(z)[\gamma_{wk}(z)\phi_{D_k}(z)]^n \gamma_k(\Lambda_{k-1})[1-\gamma_k(\Lambda_{k-1})]^n \\ &= \gamma_k(\Lambda_{k-1})\gamma_{sk}(z) \sum_{n=0}^{\infty} [(1-\gamma_k(\Lambda_{k-1}))\gamma_{wk}(z)\phi_{D_k}(z)]^n.\end{aligned}$$

From the final remark in Appendix A $(1-\gamma_k(\Lambda_{k-1}))\gamma_{wk}(z)\phi_{D_k}(z) \leq 1$. Using the expressions found for $\gamma_{wk}(z)$ and $\gamma_{sk}(z)$ in Equation (21) and Equation (22), in addition to the well known property of a geometric series,

the above equation can be re-written into

$$\begin{aligned}\phi_{R_k}(z) &= \gamma_k(\Lambda_{k-1})\gamma_{sk}(z)\frac{1}{1-[1-\gamma_k(\Lambda_{k-1})]\gamma_{wk}(z)\phi_{D_k}(z)} \\ &= \frac{(z+\Lambda_{k-1})\gamma_k(z+\Lambda_{k-1})}{z+\Lambda_{k-1}-\Lambda_{k-1}\phi_{D_k}(z)[1-\gamma_k(z+\Lambda_{k-1})]}\end{aligned} \tag{23}$$

(Conway et al., 1967, page 171).

It remains to find the Laplace transform associated with gross processing time. Conditioning on N and recalling that $P_{gk} = P_{wk} + P_{sk}$ one can, by the independence of wasted and successful processing time, write

$$\begin{aligned}\mathrm{E}\left[e^{-zP_{gk}} \mid N=n\right] &= \mathrm{E}\left[e^{-zP_{sk}}\right]\mathrm{E}\left[e^{-zP_{wk}}\right]^n \\ &= \gamma_{sk}(z)\left[\gamma_{wk}(z)\right]^n.\end{aligned}$$

Avoiding the condition on N, in the same manners as for $\phi_{R_k}(z)$ earlier, it is easy to obtain

$$\gamma_{gk}(z) = \frac{\gamma_k(z+\Lambda_{k-1})[z+\Lambda_{k-1}]}{z+\Lambda_{k-1}-\Lambda_{k-1}[1-\gamma_k(z+\Lambda_{k-1})]}.$$

It should be noted that this equals Equation (23) with $\phi_{D_k}(z) = 1$. This makes sense since gross processing time is the same as the residence time under the assumption that the class k job in question suffers no preemption.

Moments of gross processing time and residence time are needed later on. The first derivative of $\gamma_{gk}(z)$ is

$$\begin{aligned}\gamma'_{gk}(z) = &\frac{\left[\gamma_k(z+\Lambda_{k-1})+(z+\Lambda_{k-1})\gamma'_k(z+\Lambda_{k-1})\right]\left[z+\Lambda_{k-1}\gamma_k(z+\Lambda_{k-1})\right]}{\left(z+\Lambda_{k-1}-\Lambda_{k-1}[1-\gamma_k(z+\Lambda_{k-1})]\right)^2} \\ &- \frac{\left[1-\Lambda_{k-1}\gamma'_k(z+\Lambda_{k-1})\right]\left[(z+\Lambda_{k-1})\gamma_k(z+\Lambda_{k-1})\right]}{\left(z+\Lambda_{k-1}-\Lambda_{k-1}[1-\gamma_k(z+\Lambda_{k-1})]\right)^2}.\end{aligned}$$

As usual the limit is evaluated as z goes to zero. This yields the expression

$$\begin{aligned}\gamma'_{gk}(0) &= \frac{\left[\gamma_k(\Lambda_{k-1})+\Lambda_{k-1}\gamma'_k(\Lambda_{k-1})\right]-\left[1-\Lambda_{k-1}\gamma'_k(\Lambda_{k-1})\right]}{\Lambda_{k-1}\gamma_k(\Lambda_{k-1})} \\ &= (-1)\frac{1-\gamma_k(\Lambda_{k-1})}{\Lambda_{k-1}\gamma_k(\Lambda_{k-1})}.\end{aligned}$$

By Equation (50), the first moment of gross processing time is

$$\mathrm{E}\left[P_{gk}\right] = \frac{1-\gamma_k(\Lambda_{k-1})}{\Lambda_{k-1}\gamma_k(\Lambda_{k-1})} \tag{24}$$

(Conway et al., 1967, page 172). Continuing along the same path, one can obtain an expression for the second derivative of $\gamma_{gk}(z)$ as z goes to zero as

$$\gamma''_{gk}(0) = \frac{2}{\left[\Lambda_{k-1}\gamma_k(\Lambda_{k-1})\right]^2}\left[1-\gamma_k(\Lambda_{k-1})+\Lambda_{k-1}\gamma'_k(\Lambda_{k-1})\right].$$

Here

$$\begin{aligned}\gamma_k'(\Lambda_{k-1}) &= \lim_{z\to 0}\frac{\mathrm{d}}{\mathrm{d}z}\gamma_k(z+\Lambda_{k-1})\\ &= \lim_{z\to 0}\int_{p=0}^{\infty}\frac{\mathrm{d}}{\mathrm{d}z}e^{-p(z+\Lambda_{k-1})}\,\mathrm{d}G_k(p)\\ &= \lim_{z\to 0}\int_{p=0}^{\infty}-pe^{-zp}e^{-\Lambda_{k-1}p}\,\mathrm{d}G_k(p)\\ &= -\mathrm{E}\left[P_k e^{-\Lambda_{k-1}P_k}\right] \qquad (25)\end{aligned}$$

under the assumption that differentiation can be done under the integral sign (Walpole et al., 2007, page 220). Again Equation (50) yields

$$\mathrm{E}\left[P_{gk}^2\right] = \frac{2}{[\Lambda_{k-1}\gamma_k(\Lambda_{k-1})]^2}\left(1-\gamma_k(\Lambda_{k-1})-\Lambda_{k-1}\mathrm{E}\left[P_k e^{-\Lambda_{k-1}P_k}\right]\right). \qquad (26)$$

The two first moments of residence time R_k can be obtained as

$$\mathrm{E}\left[R_k\right] = (1+\Lambda_{k-1}\mathrm{E}\left[D_k\right])\,\mathrm{E}\left[P_{gk}\right] \qquad (27)$$

and

$$\begin{aligned}\mathrm{E}\left[R_k^2\right] =& (1+\Lambda_{k-1}\mathrm{E}\left[D_k\right])\,\mathrm{E}\left[P_{gk}^2\right]\\ &+ 2\Lambda_{k-1}\mathrm{E}\left[D_k\right](1+\Lambda_{k-1}\mathrm{E}\left[D_k\right])\left(\mathrm{E}\left[P_{gk}\right]\right)^2\\ &+ \Lambda_{k-1}\mathrm{E}\left[D_k^2\right]\mathrm{E}\left[P_{gk}\right]. \qquad (28)\end{aligned}$$

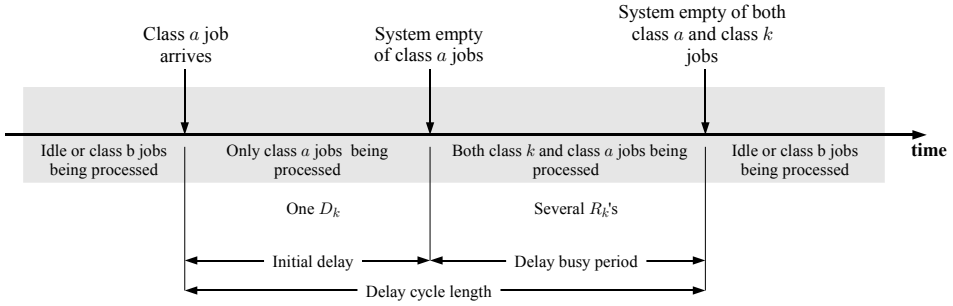
To obtain these moments is straight forward as usual, but the derivation, limit evaluation and algebraic manipulation of the resulting expressions becomes a lengthy operation. No additional insight is gained from carrying out these operations, so the above results are simply quoted directly from Conway et al. (1967, page 172).

6.4 Determination of waiting time

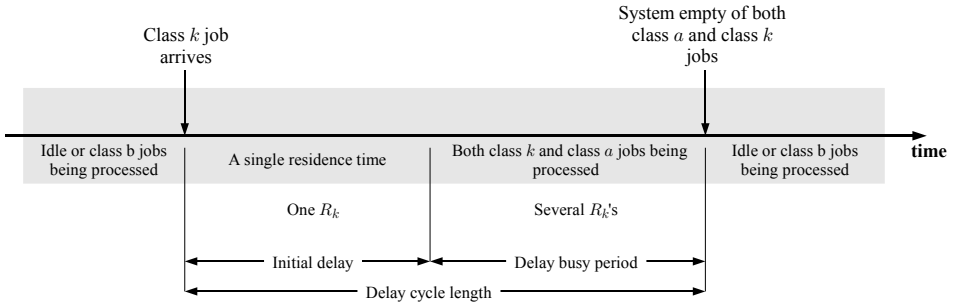
The next step in the study of the preemptive repeat-different priority discipline is to determine the Laplace transform associated with flow-time by use of the analysis of FCFS operations in delay cycles. The procedure is similar to that of Section 5.

Some class k jobs arrive while the system is idle or when the machine is processing a job of lower priority. The system is said to be in state 0 since an arriving class k job preempts the class b job being processed, if any, and goes on the machine immediately in both cases. A class k job arriving a system in state 0 suffers no waiting time as it finds the system virtually empty.

All other arriving class k jobs arrive during delay cycles of which there are now only two types, both shown schematically in Figure 5. A type a cycle



(a) Type a delay cycle



(b) Type k delay cycle

Figure 5: The two different types of delay cycles a class k customer may encounter when arriving at a busy system with preemptive repeat priority discipline.

is initiated when a class a job arrives a virtually empty system. Similarly, a type k cycle is initiated when an arriving class k job finds the system empty of both class k and class a jobs. Both cycles end when the system is emptied for both class a and class k jobs.

The Laplace transform associated with class k flow-time in a type $j = \{k, a\}$ cycle can be found from Equation (13) as

$$\phi_{F_k|j}(z) = \frac{(1 - \lambda_k \mathrm{E}[R_k])\phi_{R_k}(z)[1 - \eta_{0j}(z)]}{\mathrm{E}[T_{0j}]\,[z - \lambda_k + \lambda_k \phi_{R_k}(z)]}.$$

As discussed R_k has taken on the role of P in Equation (13) and thus $\gamma(z)$ is replaced by $\phi_{R_k}(z)$.

In the type a cycle the initial delay consists of class a jobs being processed only. Therefore the length of the initial delay is equal to the duration of a preemption and $T_{0a} = D_k$. In a type k cycle the initial delay is simply one single residence time and $T_{0k} = R_k$.

The system is said to be in state a if it is engaged in a type a cycle and in state k if it is engaged in a type k cycle. Once the steady state probabilities π_j, $j = \{0, k, a\}$, have been determined the Laplace transform associated with a class k flow-time in a preemptive repeat priority system can be found. The π_j's are obtained by considering

$$\pi_j = \frac{m_j}{l_j}$$

as in Section 5. Still m_j and l_j denote the mean length of a j cycle and the mean time between entrances to state j, respectively.

Since the expected time to the next arrival of a class a or a class k job at any time is $1/\Lambda_k$, the mean length of state 0 is given as $m_0 = 1/\Lambda_k$. This follows immediately from the fact that the system is idle, as far as a class k job is concerned, only as long as there is no class a or class k jobs present.

The mean time between two successive entrances to state 0 is the expected length of a state 0 period plus the expected length of the following cycle. When that cycle ends, the system is in state 0 once again. Conditioning on whether the state 0 period is terminated by an arriving class a or class k job, an expression for l_0 is obtained as

$$l_0 = m_0 + \frac{\Lambda_{k-1}}{\Lambda_k} m_a + \frac{\lambda_k}{\Lambda_k} m_k.$$

As already discussed $T_{0a} = D_k$ and $T_{0k} = R_k$. Hence m_a and m_k can easily be expressed as

$$m_a = \mathrm{E}\left[T_{ca}\right] = \frac{\mathrm{E}\left[D_k\right]}{1 - \lambda_k \mathrm{E}\left[R_k\right]} \quad \text{and} \quad m_k = \mathrm{E}\left[T_{ck}\right] = \frac{\mathrm{E}\left[R_k\right]}{1 - \lambda_k \mathrm{E}\left[R_k\right]}.$$

It is now a simple task to obtain π_0 as

$$\pi_0 = \frac{m_0}{l_0} = \frac{1}{1 + \Lambda_{k-1} m_a + \lambda_k m_k} = \frac{1 - \lambda_k \mathrm{E}\left[P_{rk}\right]}{1 + \Lambda_{k-1} \mathrm{E}\left[T_{bk}\right]}.$$

From the discussion of the non-preemptive priority system, only those class a and class k jobs arriving a virtually empty system initiate type a and type k cycles. Hence, the rate at which type a and type k cycles are initiated is $r_a = \Lambda_{k-1}\pi_0$ and $r_k = \lambda_k \pi_0$, respectively. From the elementary renewal theorem, Equation (55) in Appendix B, $r_j = 1/l_j$. By now all the quantities needed to calculate the steady state probabilities are known and the remaining π_j's can be obtained as

$$\pi_a = m_a r_a = \Lambda_{k-1} m_a \pi_0 = \frac{\Lambda_{k-1} \mathrm{E}\left[D_k\right]}{1 + \Lambda_{k-1} \mathrm{E}\left[D_k\right]}$$

and

$$\pi_k = m_k r_k = \lambda_k m_k \pi_0 = \frac{\lambda_k \mathrm{E}\left[R_k\right]}{1 + \Lambda_{k-1} \mathrm{E}\left[D_k\right]}.$$

The expected flow-time of a class k job is $\mathrm{E}[F_k] = \mathrm{E}[R_k] + \mathrm{E}[W_k]$. The moments of residence time are known as soon as the moments of D_k has been determined. These moments will be investigated in Section 6.5. Thus, for now it is sufficient to consider waiting time.

By conditioning on the proportion of time π_j the system is in state j, the overall waiting time can be obtained. A job arriving at the system when the system is in state 0 has no waiting time and $\phi_{W_k|0}(z) = 1$. Hence, the Laplace transform associated with a class k job's waiting time in a preemptive repeat-different priority system is

$$\begin{aligned}
\phi_{W_k}(z) &= \pi_0 + \pi_a \phi_{W_k|a}(z) + \pi_k \phi_{W_k|k}(z) \\
&= \frac{1 - \lambda_k \mathrm{E}[R_k]}{1 + \Lambda_{k-1}\mathrm{E}[D_k]} + \frac{\Lambda_{k-1}\mathrm{E}[D_k]}{1 + \Lambda_{k-1}\mathrm{E}[D_k]} \frac{(1 - \lambda_k \mathrm{E}[R_k])[1 - \phi_{D_k}(z)]}{\mathrm{E}[D_k]\,[z - \lambda_k + \lambda_k \phi_{R_k}(z)]} \\
&\quad + \frac{\lambda_k \mathrm{E}[R_k]}{1 + \Lambda_{k-1}\mathrm{E}[D_k]} \frac{(1 - \lambda_k \mathrm{E}[R_k])[1 - \phi_{R_k}(z)]}{\mathrm{E}[R_k]\,[z - \lambda_k + \lambda_k \phi_{R_k}(z)]} \\
&= \frac{1 - \lambda_k \mathrm{E}[R_k]}{1 + \Lambda_{k-1}\mathrm{E}[D_k]} \left[1 + \frac{\Lambda_{k-1}[1 - \phi_{D_k}(z)]}{\lambda_k \phi_{R_k}(z) - \lambda_k + z} + \frac{\lambda_k [1 - \phi_{R_k}(z)]}{\lambda_k \phi_{R_k}(z) - \lambda_k + z} \right] \\
&= \pi_0 \left[\frac{z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)}{\lambda_k \phi_{R_k}(z) - \lambda_k + z} \right].
\end{aligned}$$

Following the standard procedure for obtaining moments, one first differentiate $\phi_{W_k}(z)$ with respect to z, obtaining

$$\begin{aligned}
\phi'_{W_k}(z) = \pi_0 &\frac{[1 - \Lambda_{k-1}\phi'_{D_k}(z)][\lambda_k \phi_{R_k}(z) - \lambda_k + z]}{[\lambda_k \phi_{R_k}(z) - \lambda_k + z]^2} \\
&- \pi_0 \frac{[1 + \lambda_k \phi'_{R_k}(z)][z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)]}{[\lambda_k \phi_{R_k}(z) - \lambda_k + z]^2}.
\end{aligned}$$

To evaluate the limit as z tends to 0, l'Hôpital's rule must be applied twice. This yields

$$\phi'_{W_k}(0) = \lim_{z \to 0} \left[\pi_0 \frac{-\Lambda_{k-1}\phi''_{D_k}(z)[1 + \lambda_k \phi'_{R_k}(z)] - \Lambda_{k-1}\phi''_{R_k}(z)[1 - \phi'_{D_k}(z)]}{2[1 + \lambda_k \phi'_{R_k}(z)]} \right],$$

where the terms tending to 0 have been left out. Evaluating the limit finally gives

$$\mathrm{E}[W_k] = \frac{\lambda_k \mathrm{E}[R_k^2]}{2(1 - \lambda_k \mathrm{E}[R_k])} + \frac{\Lambda_{k-1}\mathrm{E}[D_k^2]}{2(1 + \Lambda_{k-1}\mathrm{E}[D_k])} \tag{29}$$

(Conway et al., 1967, page 174).

$\mathrm{E}[W_k]$ is expressed in terms of moments of R_k and D_k. In Equation (27) and Equation (28) $\mathrm{E}[R_k]$ and $\mathrm{E}[R_k^2]$ are expressed in terms of the first two moments of P_{gk} and D_k. To obtain a convenient form of $\mathrm{E}[W_k]$ expressed in terms of gross processing time and residence time, the relationship between P_{gk} and D_k must be explored.

6.5 Breakdown time

The duration of a preemption – breakdown time – of a class k job consists of the residence time of the interrupting job with higher priority and the residence time of all later arriving class $k-1$ jobs, until the system again is emptied of jobs from class 1, 2, ..., $k-1$. The breakdown times and the residence times depend on each other recursively. The residence time of jobs from the highest priority class, R_1, determines the breakdown time D_2 of jobs from the second highest priority class. Then D_2 determines the residence time of the class 2 jobs, R_2, and so on.

The breakdown time of a class $k+1$ job is either a type a cycle or a type k cycle. T_{ca} and T_{ck} denotes the length of these cycles, and

$$D_{k+1} = \begin{cases} T_{ca} & \text{with probability } \Lambda_{k-1}/\Lambda_k \\ T_{ck} & \text{with probability } \lambda_k/\Lambda_k, \end{cases}$$

since the job causing the preemption is from class a with probability Λ_{k-1}/Λ_k and from class k with probability λ_k/Λ_k.

The first moment of T_{ca} and T_{ck} is, as already discussed,

$$\mathrm{E}\left[T_{ca}\right] = \frac{\mathrm{E}\left[D_k\right]}{1-\lambda_k \mathrm{E}\left[R_k\right]} \quad \text{and} \quad \mathrm{E}\left[T_{ck}\right] = \frac{\mathrm{E}\left[R_k\right]}{1-\lambda_k \mathrm{E}\left[R_k\right]},$$

respectively. The second moments can be found from Equation (12) in Section 4 as

$$\mathrm{E}\left[T_{ca}^2\right] = \frac{\lambda_k \mathrm{E}\left[R_k^2\right]}{(1-\lambda_k \mathrm{E}\left[R_k\right])^3} \mathrm{E}\left[D_k\right] + \frac{\mathrm{E}\left[D_k^2\right]}{(1-\lambda_k \mathrm{E}\left[R_k\right])^2}$$

and

$$\mathrm{E}\left[T_{ck}^2\right] = \frac{\lambda_k \mathrm{E}\left[R_k^2\right]}{(1-\lambda_k \mathrm{E}\left[R_k\right])^3} \mathrm{E}\left[R_k\right] + \frac{\mathrm{E}\left[R_k\right]^2}{(1-\lambda_k \mathrm{E}\left[R_k\right])^2}.$$

Conditioning on type of cycle the first moment of D_{k+1} is obtained easily as

$$\mathrm{E}\left[D_{k+1}\right] = \frac{\Lambda_{k-1}\mathrm{E}\left[D_k\right] + \lambda_k \mathrm{E}\left[R_k\right]}{\Lambda_k(1-\lambda_k \mathrm{E}\left[R_k\right])}. \tag{30}$$

Similarly, the second moment of D_{k+1} is

$$\begin{aligned} \mathrm{E}\left[D_{k+1}^2\right] &= \frac{\Lambda_{k-1}}{\Lambda_k}\mathrm{E}\left[T_{ca}^2\right] + \frac{\lambda_k}{\Lambda_k}\mathrm{E}\left[T_{ck}^2\right] \\ &= \frac{\Lambda_{k-1}\mathrm{E}\left[D_k^2\right](1-\lambda_k \mathrm{E}\left[R_k\right]) + \lambda_k \mathrm{E}\left[R_k^2\right](1+\Lambda_{k-1}\mathrm{E}\left[D_k\right])}{\Lambda_k(1-\lambda_k \mathrm{E}\left[R_k\right])^3} \end{aligned} \tag{31}$$

(Conway et al., 1967, page 174).

Cho and Un (1993, page 140) obtained much simpler expressions for the first two moments of D_k. As class 1 jobs don't suffer preemptions $D_1 = 0$. From Equation (30), $\mathrm{E}[D_2]$ can be found to be

$$\mathrm{E}[D_2] = \frac{\lambda_1 \mathrm{E}[R_1]}{\lambda_1(1 - \lambda_1 \mathrm{E}[R_1]}.$$

A second consequence of class 1 jobs not suffering preemptions is that the residence time is simply gross processing time; $\mathrm{E}[R_1] = \mathrm{E}[P_{g1}]$.

Using $\tilde{\rho}_k = \lambda_k \mathrm{E}[P_{gk}]$ and introducing $\sigma_k = \sum_{j=1}^{k} \tilde{\rho}_j$ the above expression for $\mathrm{E}[D_2]$ can be written in the form

$$\mathrm{E}[D_k] = \frac{\sigma_{k-1}}{\Lambda_{k-1}(1 - \sigma_{k-1})}. \tag{32}$$

Using an inductive argument (Drekic and Stanford, 2000, page 293–294) this can be proven to hold for all $k = 1, 2, \ldots, K$.

From Equation (27)

$$\mathrm{E}[R_l] = (1 + \Lambda_{l-1}\mathrm{E}[D_l])\mathrm{E}[P_{gl}]\,.$$

Assuming Equation (32) holds for all classes up to $k-1$ and inserting that into the above expression yields

$$\begin{aligned}
\mathrm{E}[R_{k-1}] &= (1 + \lambda_{k-1}\mathrm{E}[D_{k-1}])\mathrm{E}[P_{gk}] \\
&= (1 + \frac{\Lambda_{k-2}\sigma_{k-2}}{\Lambda_{k-2}(1 - \sigma_{k-2})})\mathrm{E}[P_{g,k-1}] \\
&= \frac{\mathrm{E}[P_{g,k-1}]}{1 - \sigma_{k-2}}.
\end{aligned}$$

Finally, inserting the expressions found for $\mathrm{E}[R_{k-1}]$ and $\mathrm{E}[D_{k-1}]$ into Equation (30) gives

$$\begin{aligned}
\mathrm{E}[D_k] &= \frac{\frac{\Lambda_{k-2}\sigma_{k-2}}{\Lambda_{k-2}(1-\sigma_{k-2})} + \frac{\lambda_{k-1}\mathrm{E}[P_{g,k-1}]}{1-\sigma_{k-2}}}{\Lambda_{k-1}\left(1 - \frac{\lambda_{k-1}\mathrm{E}[P_{g,k-1}]}{1-\sigma_{k-2}}\right)} \\
&= \frac{\sigma_{k-2} + \tilde{\rho}_{k-1}}{\Lambda_{k-1}\left(1 - \sigma_{k-2} - \tilde{\rho}_{k-1}\right)} \\
&= \frac{\sigma_{k-1}}{\Lambda_{k-1}(1 - \sigma_{k-1})},
\end{aligned}$$

which is exactly Equation (32) and thereby inductively proves that it holds.

A similar approach leads to a simplification of the second moment of breakdown time. The expressions involved (Equation (31) and Equation (33)) are rather complicated in form and the inductive proof becomes both lengthy and ugly. As it is simply an algebraic operation providing no additional

information about the problem, the final result – obtained by Cho and Un (1993, page 140) and studied closer by Drekic and Stanford (2000, page 294) – is simply stated:

$$\mathrm{E}\left[D_k^2\right] = \frac{1}{\Lambda_{k-1}(1-\sigma_{k-1})^2} \sum_{j=1}^{k-1} \lambda_j \frac{(1-\sigma_{j-1})^2}{1-\sigma_j} \mathrm{E}\left[R_j^2\right]. \tag{33}$$

6.6 Calculation of flow-time

The first and second moment of gross processing time are known through Equation (24) and Equation (26). Since the class 1 jobs cannot be preempted, the class 1 breakdown time is zero. Hence the first and second moment of R_1 can be obtained from Equation (27) and Equation (28).

Next, the two first moments of D_k and R_k can be obtained recursively for $k = 2, 3, \ldots, K$.

Finally, all quantities needed for calculating the expected flow-times are at hand. Flow time is residence time plus waiting time. Using Equation (29), the first moment of the class k flow-time is

$$\mathrm{E}\left[F_k\right] = \mathrm{E}\left[R_k\right] + \frac{\lambda_k \mathrm{E}\left[R_k^2\right]}{2(1-\lambda_k \mathrm{E}\left[R_k\right])} + \frac{\Lambda_{k-1} \mathrm{E}\left[D_k^2\right]}{2(1+\Lambda_{k-1} \mathrm{E}\left[D_k\right])}. \tag{34}$$

7 The combined priority discipline

Typical in classic queuing theory is to let preemptions be allowed in the entire queuing system or not at all. Cho and Un (1993) studied a combined preemptive/non-preemptive priority discipline – introduced for a two-class system by Jaiswal (1968) – allowing preemptions during the early stage of processing only. They studied the effect of combining the preemptive-repeat and the preemptive resume-equal disciplines with the non-preemptive discipline. Drekic and Stanford (2000) extended the theory to also apply to the preemptive repeat-different priority rule.

The above mentioned papers consider several threshold policies defining when preemptions are allowed and when they are prevented. One possibility is to allow preemptions until a specified amount of processing time has been completed. Another is to allow preemptions when more than a certain amount of processing time remains before completion.

However, the threshold policy of interest here is that which Drekic and Stanford (2000) refers to as proportion-based. The idea is to protect the remaining processing time from further preemptions once a certain proportion φ, $0 \leq \varphi \leq 1$, of the total processing time has been successfully completed. In the remaining, protected processing interval even higher prioritized jobs arriving at the system must wait for the current job to be completed.

7.1 Characteristic properties and notation

It is still class k jobs that are of interest. As before class a jobs denote jobs from all classes with priority above the class k jobs. Similarly class b jobs are jobs from classes with priority below the class k jobs.

The system of interest is not work conserving. The class k utilization factor is

$$\tilde{\rho}_k = \lambda_k \mathrm{E}\left[P_{gk}\right].$$

Recall from Section 6 that $\Lambda_k = \sum_{j=1}^{k} \lambda_j$ and $\sigma_k = \sum_{j=1}^{k} \tilde{\rho}_j$.

The threshold defining the part of the processing interval where preemptions are allowed and the part where preemptions are prevented can be class dependent. Therefore, a class k job can be preempted by a higher prioritized job until it has completed $\varphi_k P_k$ of its total processing time. The remaining processing interval $(1-\varphi_k)P_k$ is regarded as protected and the system becomes non-preemptive once the threshold is reached.

The protected processing interval will be denoted by $S_k = (1-\varphi_k)P_k$. It is easy to confirm that the random variable S_k has ith moment given as $(1-\varphi_k)^i \mathrm{E}\left[P_k^i\right]$ since φ_k is a constant known for each class. It should also be noted that the Laplace transform associated with the protected class k processing interval is

$$\phi_{S_k}(z) = \gamma_k[(1-\varphi_k)z].$$

Under the combined priority discipline also low priority jobs can keep the jobs of higher priority from starting their processing. When a class b job complete the part of its processing where preemptions are possible before a class k or class a job arrives, the job of higher priority must wait during the protected part of the class b processing interval.

To handle this deviation from the pure preemptive priority discipline the completion time of a class k job must be introduced. Completion time is defined as residence time (still denoted by R_k for class k) plus the amount of time, if any, until the machine is ready to process the next class k job. The completion time of a class k job is denoted by C_k.

$C_k > R_k$ when at least one higher prioritized job, from class a, is forced to wait during a protected, non-preemptive processing interval. C_k takes on the role of effective processing time, since the next class k job is forced to wait for the jobs of higher priority to be processed, even though the previous class k job was allowed to finish.

D_k still denotes the class k breakdown time – the time between a class k preemption and the next processing attempt.

Both class a and class k jobs see the system as virtually empty when the machine is either idle or processing the preemptive part of a class b job's processing time. Their processing starts immediately upon arrival in these situations.

A class k job may arrive at the system while the machine is virtually idle, or during one of three different types of delay cycles, all shown schematically in Figure 6.

The type a cycles are initiated when a class a job arrives at a virtually idle system. Hence, the initial delay has the length of a breakdown time D_k and the delay busy period consists of several class k completion times.

A type k cycle is really just a busy period where each job has effective processing time C_k.

The third type of delay cycle is called a type b_i cycle. It is initiated when a class b job reaches its threshold and is protected from further preemptions. Class b is formed by class i jobs, $i = k+1,\ k+2,\ \ldots,\ K$. Each of these classes can have different thresholds φ_i. As a consequence, all the type b_i cycles do not necessarily have the same initial delays. Analogous to the non-preemptive case, the initial delay in a b_i cycle is itself a delay cycle.

The discussion of the delay cycles is continued after the breakdown times in a combined preemptive/non-preemptive priority system has been explored.

7.2 Breakdown time revisited

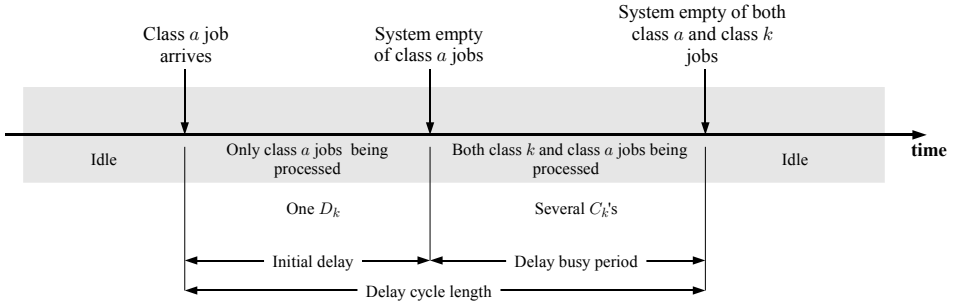
It is now possible to make some important modifications to earlier results regarding breakdown times. However, a link between the gross processing time and completion time is needed and must be accounted for first.

A class k completion time consists of one class k gross processing time and a random number of busy periods caused by arrivals of higher priority jobs. All class a jobs arriving during P_{gk} are responsible for a busy period with the duration of a breakdown time D_k. Those class a jobs arriving during the early stage of the class k process causes preemptions. Those that arrive after the class k threshold is reached causes busy periods with the length of breakdown times after the class k job is completed. The number of high priority jobs arriving during the gross processing time P_{gk} is given by a Poisson process with rate Λ_{k-1}. It follows that

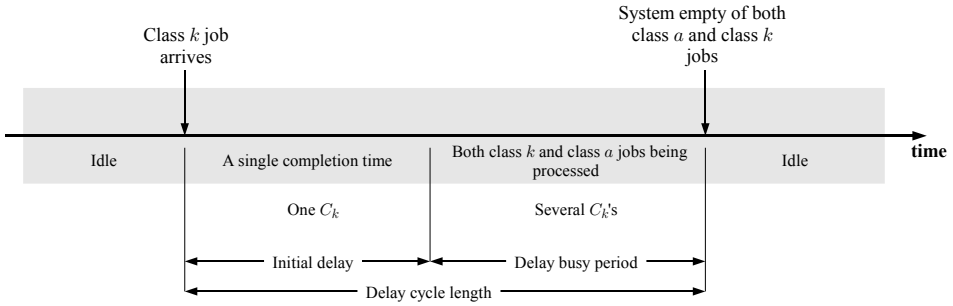
$$\mathrm{E}\left[C_k\right] = (1 + \Lambda_{k-1}\mathrm{E}\left[D_k\right])\mathrm{E}\left[P_{gk}\right] \tag{35}$$

(Drekic and Stanford, 2000, page 293). The similarity between this expression and Equation (27) connecting residence time and gross processing time in Section 6 is obvious.

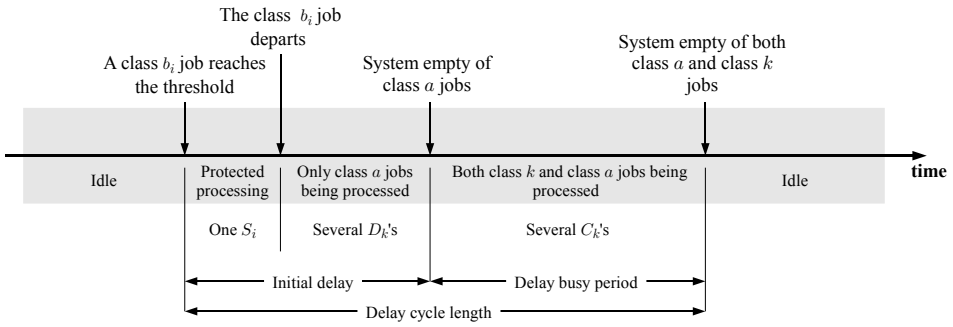
As under the fully preemptive priority discipline, the breakdown time of a class $k+1$ customer is a type a cycle or a type k cycle with probability Λ_{k-1}/Λ_k and λ_k/Λ_k, respectively. From Figure 6 it is clear that the delay busy periods in both cycles consists of completion times and not residence times as was the case in Section 6. It is straight forward to obtain the mean



(a) Type a delay cycle



(b) Type k delay cycle



(c) Type b_i delay cycle

Figure 6: Different types of delay cycles in a system with combined preemptive/non-preemptive priority discipline.

length of a type a cycle as

$$m_a = \frac{\mathrm{E}\left[D_k\right]}{1 - \lambda_k \mathrm{E}\left[C_k\right]}$$

and the mean length of a type k cycle as

$$m_k = \frac{\mathrm{E}\left[C_k\right]}{1 - \lambda_k \mathrm{E}\left[C_k\right]}$$

from Equation (11) and Equation (2), respectively.

Conditioning on their occurrence a recursive expression for the first moment of breakdown time can be obtained as

$$\mathrm{E}\left[D_{k+1}\right] = \frac{\Lambda_{k-1}\mathrm{E}\left[D_k\right] + \lambda_k \mathrm{E}\left[C_k\right]}{\Lambda_k(1 - \lambda_k \mathrm{E}\left[C_k\right])}.$$

In a similar fashion the second moment becomes

$$\mathrm{E}\left[D_{k+1}^2\right] = \frac{\Lambda_{k-1}\mathrm{E}\left[D_k^2\right](1 - \lambda_k \mathrm{E}\left[C_k\right]) + \lambda_k \mathrm{E}\left[C_k^2\right](1 + \Lambda_{k-1}\mathrm{E}\left[D_k\right])}{\Lambda_k(1 - \lambda_k \mathrm{E}\left[C_k\right])^3}.$$

Class 1 jobs are never preempted and $D_1 = 0$. Using Equation (35) and an otherwise identical inductive argument to that of Section 6.5 it is easy to obtain a simplified expressions for the first moment of breakdown time as

$$\mathrm{E}\left[D_k\right] = \frac{\sigma_{k-1}}{\Lambda_{k-1}(1 - \sigma_{k-1})}. \tag{36}$$

This is identical to the expression sated in Equation (32).

An expression for the second moment of breakdown time becomes

$$\mathrm{E}\left[D_k^2\right] = \frac{1}{\Lambda_{k-1}(1 - \sigma_{k-1})^2} \sum_{j=1}^{k-1} \lambda_j \frac{(1 - \sigma_{j-1})^2}{1 - \sigma_j} \mathrm{E}\left[C_j^2\right]$$

(Drekic and Stanford, 2000, page 294). This is similar to Equation (33), but residence time has been replaced by completion time.

7.3 Waiting time

As usual the mean length of a type j delay cycle is denoted by m_j. The exact expressions for the type j cycle can easily be found by considering Equation (11). However, it is not necessary to consider these expressions at this point as the final result for waiting time does not depend om them directly.

From Equation (14) the conditional Laplace transform associated with waiting time in a type a cycle is

$$\phi_{W_k|a}(z) = \frac{1 - \phi_{D_k}(z)}{m_a[z - \lambda_k + \lambda_k \phi_{C_k}(z)]},$$

since Laplace transform associated with initial delay is $\phi_{D_k}(z)$ and the Laplace transform associated with the delay busy period is $\phi_{C_k}(z)$.

The conditional Laplace transform associated with waiting time in a type k cycle is

$$\phi_{W_k|k}(z) = \frac{1 - \phi_{C_k}(z)}{m_k[z - \lambda_k + \lambda_k \phi_{C_k}(z)]},$$

since $\phi_{C_k}(z)$ is associated with both initial delay and the delay busy period in this case.

For the type b_i cycle, $i = k+1, k+2, \ldots, K$, one find

$$\phi_{W_k|b_i}(z) = \frac{1 - \eta_{0,b_i}(z)}{m_{b_i}[z - \lambda_k + \lambda_k \phi_{C_k}(z)]}.$$

To determine the initial delay $\eta_{0,b_i}(z)$ in a type b_i cycle, observe that class a jobs arrive at rate Λ_{k-1} during the protected processing interval $S_i = (1 - \varphi_i)P_i$. Each of these arrivals induces a busy period of duration D_k. It follows that

$$\eta_{0,b_i}(z) = \phi_{S_i}[z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)]$$

from Equation (10) since the initial delay of a b_i cycle is itself a delay cycle (see Figure 6c).

It now remains to find the proportion of arriving class k jobs that find the system engaged in the different types of cycles before the unconditional Laplace transform associated with waiting time can be obtained. As before, this is done by considering the steady state probabilities.

The system is said to be in state 0 if it is virtually empty from a class k job's point of view. This is the situation if the machine is idle or if a class b job is in the non-protected part of its processing interval. Thus the steady state probability of state 0 is

$$\pi_0 = 1 - \sigma_k - \sum_{i=k+1}^{K} \lambda_i \mathrm{E}\left[S_i\right].$$

As in Section 6 the system is in state a when it is engaged in a type a cycle. Only those class a jobs arriving at a virtually empty system initiate a type a cycle. The rate at which the system enters state a is thus $r_a = \Lambda_{k-1}\pi_0$. From the elementary renewal theorem, Equation (55) in Appendix B, and Equation (18)

$$\pi_a = \Lambda_{k-1}\pi_0 m_a.$$

An analogous argument yields

$$\pi_k = \Lambda_{k-1}\pi_0 m_k$$

for state k.

All queuing systems considered in this text are assumed to be non-saturated. That implies that all $i = \{k+1, \ldots, K\}$ jobs initiate a type

Notation
$A(z) = z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)$
$A'(z) = 1 - \Lambda_{k-1}\phi'_{D_k}(z)$
$A''(z) = -\Lambda_{k-1}\phi''_{D_k}(z)$
$B_i(z) = \lambda_i(1 - \phi_{S_i}[z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)])$
$B'_i(z) = -\lambda_i(\phi'_{S_i}[z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)])[1 - \Lambda_{k-1}\phi'_{D_k}(z)]$
$B''_i(z) = -\lambda_i(\phi''_{S_i}[z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)])[1 - \Lambda_{k-1}\phi'_{D_k}(z)]^2$ $+\lambda_i(\phi'_{S_i}[z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)])[\Lambda_{k-1}\phi''_{D_k}(z)]$
$C(z) = z - \lambda_k + \lambda_k\phi_{C_k}(z)$
$C'(z) = 1 + \lambda_k\phi'_{C_k}(z)$
$C''(z) = \lambda_k\phi''_{C_k}(z)$

Table 1: Compact notation used in the derivation of $\mathrm{E}[W_k]$

b_i cycle eventually. A class i job arrives whit arrival rate λ_i. The proportion of time the system is engaged in a type b_i cycle is thus

$$\pi_{b_i} = \lambda_i m_{b_i}.$$

This is mainly the same argument used for type b cycles under the non-preemptive priority discipline in Section 5.

The unconditional Laplace transform associated with class k waiting time in a combined preemptive/non-preemptive priority system can now be obtained as

$$\phi_{W_k}(z) = \pi_0 + \pi_a\phi_{W_k|a}(z) + \pi_k\phi_{W_k|k}(z) + \pi_{b_i}\phi_{W_k|b_i}(z),$$

since the waiting time is zero for a class k job arriving at an virtually empty system and hence $\phi_{W_k|0}(z) = 1$. Simplification of the above equation yields

$$\begin{aligned}\phi_{W_k}(z) = {} & \frac{\pi_0[z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)]}{z - \lambda_k - \lambda_k\phi_{C_k}(z)} \\ & + \frac{\sum_{i=k+1}^{K}\lambda_i(1 - \phi_{S_i}[z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)])}{z - \lambda_k - \lambda_k\phi_{C_k}(z)}. \end{aligned} \tag{37}$$

As before the first moment of waiting time is of interest. Differentiation of $\phi_{W_k}(z)$ yields a very long expression, and the operation is made more manageable by introducing the notation in Table 1.

Now Equation (37) can be rewritten to

$$\phi_{W_k}(z) = \pi_0\frac{A(z)}{C(z)} + \frac{\sum_{i=k+1}^{K}B_i(z)}{C(z)}.$$

$\phi'_{W_k}(z)$ can thus be written relatively compact as

$$\phi'_{W_k}(z) = \pi_0 \left(\frac{A'(z)C(z) - A(z)C'(z)}{[C(z)]^2} \right) + \frac{\sum_{i=k+1}^{K} \left(B'_i(z)C(z) - B_i(z)C'(z) \right)}{[C(z)]^2}.$$

$A(z)$, $B_i(z)$ and $C(z)$ all tend to zero as z approaches 0, and it follows that l'Hôpital's rule must be applied to evaluate the limit. In fact it is necessary to apply l'Hôpital's rule twice. Differentiating the numerator and denominator of the above expression for $\phi'_{W_k}(z)$ twice, leaving out those terms still tending to zero, yields

$$\phi'_{W_k}(0) = \lim_{z \to 0} \left\{ \pi_0 \frac{A''(z)C'(z) - A'(z)C''(z)}{2[C'(z)]^2} + \frac{\sum_{i=k+1}^{K} [B''_i(z)C'(z) - B'_i(z)C''(z)]}{2[C'(z)]^2} \right\}. \tag{38}$$

Evaluating the involved expressions in the limit as $z \to 0$ gives

$$\begin{aligned} A'(0) &= 1 + \Lambda_{k-1} \mathrm{E}\left[D_k\right], \\ A''(0) &= -\Lambda_{k-1} \mathrm{E}\left[D_k^2\right], \\ B'_i(0) &= \lambda_i \mathrm{E}\left[S_i\right] (1 + \Lambda_{k-1} \mathrm{E}\left[D_k\right]), \\ B''_i(0) &= -\lambda_i \mathrm{E}\left[S_i^2\right] (1 + \Lambda_{k-1} \mathrm{E}\left[D_k\right])^2 - \lambda_i \mathrm{E}\left[S_i\right] \Lambda_{k-1} \mathrm{E}\left[D_k^2\right], \\ C'(0) &= 1 - \lambda_k \mathrm{E}\left[C_k\right] \end{aligned}$$

and

$$C''(0) = \lambda_k \mathrm{E}\left[C_k^2\right].$$

Inserting these limits together with $\pi_0 = 1 - \sigma_k - \sum_{i=k+1}^{K} \lambda_i \mathrm{E}\left[S_i\right]$ into Equation (38), taking into account that $\mathrm{E}\left[W_k\right] = -\phi'_{W_k}(0)$, leads to

$$\begin{aligned} \mathrm{E}\left[W_k\right] = {} & (1 - \sigma_k) \frac{\left[\Lambda_{k-1} \mathrm{E}\left[D_k^2\right] (1 - \lambda_k \mathrm{E}\left[C_k\right])\right]}{2(1 - \lambda_k \mathrm{E}\left[C_k\right])^2} \\ & + (1 - \sigma_k) \frac{\left[\lambda_k \mathrm{E}\left[C_k^2\right] (1 + \Lambda_{k-1} \mathrm{E}\left[D_k\right])\right]}{2(1 - \lambda_k \mathrm{E}\left[C_k\right])^2} \\ & + \sum_{i=k+1}^{K} \lambda_i \mathrm{E}\left[S_i^2\right] \frac{\left[(1 + \Lambda_{k-1} \mathrm{E}\left[D_k\right])^2 (1 - \lambda_k \mathrm{E}\left[C_k\right])\right]}{2(1 - \lambda_k \mathrm{E}\left[C_k\right])^2}. \end{aligned}$$

Using Equation (35) and Equation (36) it is easy to find that

$$1 + \Lambda_{k-1} \mathrm{E}\left[D_k\right] = 1/(1 - \sigma_{k-1})$$

and

$$1 - \lambda_k \mathrm{E}\left[C_k\right] = (1 - \sigma_k)/(1 - \sigma_{k-1}).$$

The expression for expected waiting time can now be simplified to

$$\mathrm{E}\left[W_k\right] = (1-\sigma_{k-1})\frac{\Lambda_{k-1}\mathrm{E}\left[D_k^2\right]}{2} + \frac{\lambda_k \mathrm{E}\left[C_k^2\right] + \sum_{i=k+1}^{K}\lambda_i \mathrm{E}\left[S_i^2\right]}{(1-\sigma_{k-1})(1-\sigma_k)}. \tag{39}$$

This final expression for expected class k waiting time under the combined preemptive/non-preemptive priority discipline is similar to the expression obtained by Cho and Un (1993, page 137). Due to what must be a misprint, the expression obtained by Drekic and Stanford (2000, page 297) does not have the factor $1-\sigma_{k-1}$ in its first term. In the next section it will be shown that Equation (39) fits simulations of the train problem very well.

Before that, expressions for completion time, gross processing time and residence time under the combined priority discipline must be discussed.

7.4 Waiting time components

In order to calculate class k waiting time, and hence also flow-time, expressions for residence time, gross processing time and completion time must be explored. Their Laplace transforms will be obtained following the procedure suggested by Cho and Un (1993, page 134–135), but extended to hold for the repeat-different discipline. The moments of interest will simply be stated. The derivation of the moments are extremely time consuming and of little interest to the reader, who probably has seen more than enough differentiation of Laplace transforms by now.

Analogous to Section 6 wasted and successful class k processing time will be denoted by P_{wk} and P_{sk}, respectively. The number of preemptions suffered by a class k job is denoted by N. The residence time R_k of a class k job suffering N preemptions under the combined priority discipline is the sum of N wasted processing times and breakdown times plus one successful processing time.

Of the class k processing time the preemptible portion is $\varphi_k P_k$ and the non-preemptible portion is $(1-\varphi_k)P_k$. A preemption occurs when a class a job arrives at the system during the preemptible part of the class k processing time. Class a jobs arrive with rate Λ_{k-1}. As in Section 6.2, let Y denote the time interval from the start of a given processing attempt to the arrival of a class a job. The probability of a given processing attempt being successful is the probability of Y being greater, or equal to, $\varphi_k P_k$;

$$\Pr\left(Y \geq \varphi_k P_k\right) = \int_{p=0}^{\infty}\int_{y=\varphi_k p}^{\infty} \Lambda_{k-1} e^{-\Lambda_{k-1}y}\mathrm{d}y\mathrm{d}G_k(p) = \gamma_k(\Lambda_{k-1}\varphi_k).$$

It follows that the number of preemptions suffered by a class k job is geometrically distributed with parameter $\gamma_k(\Lambda_{k-1}\varphi_k)$ as

$$\Pr\left(N=n\right) = \left[1-\gamma_k(\Lambda_{k-1}\varphi_k)\right]^n \gamma_k(\Lambda_{k-1}\varphi_k).$$

Analogous to Equation (21) and Equation (22) the Laplace transforms associated with wasted and successful processing time become

$$\gamma_{wk}(z) = \frac{\Lambda_{k-1}}{z + \Lambda_{k-1}} \frac{1 - \gamma_k\left[(z + \Lambda_{k-1})\varphi_k\right]}{1 - \gamma_k(\Lambda_{k-1}\varphi_k)}$$

and

$$\gamma_{sk}(z) = \frac{1 - \gamma_k\left(z + \Lambda_{k-1}\varphi_k\right)}{\gamma_k(\Lambda_{k-1}\varphi_k)},$$

respectively.

Pursuing the same path leading to Equation (23) in Section 6.3 now leads to the Laplace transform associated with residence time under the combined priority rule.

$$\phi_{R_k}(z) = \frac{(z + \Lambda_{k-1})\gamma_k(z + \Lambda_{k-1}\varphi_k)}{z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)[1 - \gamma_k([z + \Lambda_{k-1}]\varphi_k)]}, \tag{40}$$

which was also obtained by Drekic and Stanford (2000, page 300).

Since gross processing time equals the residence time under the assumption that the class k job suffers no preemption, the Laplace transform associated with P_{gk} can be found directly from Equation (40) simply by letting $\phi_{D_k}(z) = 1$. This yields

$$\gamma_{gk}(z) = \frac{(z + \Lambda_{k-1})\gamma_k(z + \Lambda_{k-1}\varphi_k)}{z + \Lambda_{k-1} - \Lambda_{k-1}[1 - \gamma_k([z + \Lambda_{k-1}]\varphi_k)]}. \tag{41}$$

The completion time is the residence time plus the delay busy period caused by all the class a jobs that arrived during the non-preemptible portion of the class k job's processing time. This delay busy period consists of several busy periods, each with duration D_k. Since the class a jobs causing the busy periods accumulated during an interval of length $(1 - \varphi_k)P_k$, it is clear that the delay cycle of interest has initial delay associated with the Laplace transform

$$\eta_0(z) = \gamma_k[(1 - \varphi_k)z].$$

The initial delay of the cycle is already accounted for in the Laplace transform associated with residence time. By Equation (9) in Section 4 the Laplace transform associated with the delay busy period is given as

$$\eta_b(z) = \gamma_k([1 - \varphi_k][\Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)]).$$

Now, the Laplace transform associated with completion time can be written

$$\phi_{C_k}(z) = \phi_{R_k}(z)\gamma_k([1 - \varphi_k][\Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)]).$$

Observing that

$$\gamma_k(z + \Lambda_{k-1}\varphi_k)\gamma_k(\Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z) - \Lambda_{k-1}\varphi_k - \Lambda_{k-1}\phi_{D_k}(z)\varphi_k)$$
$$= \gamma_k[z + \Lambda_{k-1} - \Lambda_{k-1}\phi_{D_k}(z)(1 - \varphi_k)]$$

by the convolution property, the Laplace transform associated with completion time is

$$\phi_{C_k}(z) = \frac{(z+\Lambda_{k-1})\gamma_k[z+\Lambda_{k-1}-\Lambda_{k-1}\phi_{D_k}(z)(1-\varphi_k)]}{z+\Lambda_{k-1}-\Lambda_{k-1}\phi_{D_k}(z)[1-\gamma_k([z+\Lambda_{k-1}]\varphi_k)]} \tag{42}$$

(Drekic and Stanford, 2000, page 300)

Derivation and limit evaluation of Equation (40), Equation (41) and Equation (42) yields the moments of the random variables R_k, P_{gk} and C_k, respectively. The task involves a lot of tedious algebra. Not to test the reader's patience, the moments of interest are simply stated as they are obtained by Drekic and Stanford (2000, page 311):

$$\begin{aligned}
\mathrm{E}[R_k] &= \frac{(1+\Lambda_{k-1}\mathrm{E}[D_k])(1-\gamma_k(\Lambda_{k-1}\varphi_k))}{\Lambda_{k-1}\gamma_k(\Lambda_{k-1}\varphi_k)} \\
&\quad - \frac{\Lambda_{k-1}(1-\varphi_k)\gamma_k'(\Lambda_{k-1}\varphi_k)}{\Lambda_{k-1}\gamma_k(\Lambda_{k-1}\varphi_k)}, \quad k \geq 2. \\
\mathrm{E}[P_{gk}] &= \frac{1-\gamma_k(\Lambda_{k-1}\varphi_k)-\Lambda_{k-1}(1-\varphi_k)\gamma_k'(\Lambda_{k-1}\varphi_k)}{\Lambda_{k-1}\gamma_k(\Lambda_{k-1}\varphi_k)}, \quad k \geq 2. \\
\mathrm{E}\left[P_{gk}^2\right] &= \frac{(1-\varphi_k^2)\gamma_k''(\Lambda_{k-1}\varphi_k)}{\gamma_k(\Lambda_{k-1}\varphi_k)} \\
&\quad + \frac{2[1-\Lambda_{k-1}(1-\varphi_k)\gamma_k'(\Lambda_{k-1}\varphi_k)]}{[\Lambda_{k-1}\gamma_k(\Lambda_{k-1}\varphi_k)]^2} \\
&\quad \times (1-\gamma_k(\Lambda_{k-1}\varphi_k)+\Lambda_{k-1}\varphi_k\gamma_k'(\Lambda_{k-1}\varphi_k)), \quad k \geq 2. \\
\mathrm{E}[C_k] &= (1+\Lambda_{k-1}\mathrm{E}[D_k])\mathrm{E}[P_{gk}], \quad k \geq 2. \\
\mathrm{E}\left[C_k^2\right] &= (1+\Lambda_{k-1}\mathrm{E}[D_k])\mathrm{E}\left[P_{gk}^2\right] + \Lambda_{k-1}\mathrm{E}\left[D_k^2\right]\mathrm{E}[P_{gk}] \\
&\quad + 2\Lambda_{k-1}\mathrm{E}[D_k](1+\Lambda_{k-1}\mathrm{E}[D_k]) \\
&\quad \times \left[\frac{\mathrm{E}[P_{gk}][1-\gamma_k(\Lambda_{k-1}\varphi_k)]}{\Lambda_{k-1}\gamma_k(\Lambda_{k-1}\varphi_k)} + \frac{(1-\varphi_k)^2\gamma_k''(\Lambda_{k-1}\varphi_k)}{2\gamma_k(\Lambda_{k-1}\varphi_k)}\right], \quad k \geq 2.
\end{aligned}$$

Here, by similar arguments to those used to obtain Equation (25) in Section 6.3,

$$\begin{aligned}
\gamma_k(\Lambda_{k-1}\varphi_k) &= \mathrm{E}\left[e^{\Lambda_{k-1}\varphi_k P_k}\right], \\
\gamma_k'(\Lambda_{k-1}\varphi_k) &= -\mathrm{E}\left[P_k e^{\Lambda_{k-1}\varphi_k P_k}\right],
\end{aligned}$$

and

$$\gamma_k''(\Lambda_{k-1}\varphi_k) = \mathrm{E}\left[P_k^2 e^{\Lambda_{k-1}\varphi_k P_k}\right].$$

The expressions above are valid for priority classes $k = 2, \ldots, K$. Priority class 1 suffers no preemptions. It is clear that $D_k = 0$. No preemptions also implies that gross processing time, residence time and completion time all are equal to the class 1 processing time. Thus, for $k = 1$,

$$\mathrm{E}[P_{g1}] = \mathrm{E}[R_1] = \mathrm{E}[C_1] = \mathrm{E}[P_1]$$

and

$$\mathrm{E}\left[P_{g1}^2\right] = \mathrm{E}\left[R_1^2\right] = \mathrm{E}\left[C_1^2\right] = \mathrm{E}\left[P_1^2\right].$$

7.5 Expected flow-time and its limits

From the recursive connection between completion times and breakdown times discussed in Section 7.2, the expected flow-time can now be calculated for all classes $k = 1, \ldots, K$ by the formula

$$\begin{aligned}\mathrm{E}\left[F_k\right] = \mathrm{E}\left[R_k\right] & \\ + (1-\sigma_{k-1})\frac{\Lambda_{k-1}\mathrm{E}\left[D_k^2\right]}{2} & \\ + \frac{\lambda_k \mathrm{E}\left[C_k^2\right] + \sum_{i=k+1}^{K}\lambda_i \mathrm{E}\left[S_i^2\right]}{(1-\sigma_{k-1})(1-\sigma_k)}. & \end{aligned} \tag{43}$$

This is, as mentioned in the introduction, the core result of this study.

The combined preemptive/non-preemptive priority system reduces to a strictly non-preemptive system in the case of $\varphi_k = 0$, for $k = 1, \ldots, K$. In the case of $\varphi_k = 1$ for for $k = 1, \ldots, K$, the combined system becomes a strictly preemptive system. That is, Equation (43) becomes Equation (19) and Equation (34) for each of the two threshold extrema, respectively.

This is illustrated for a system with 4 priority classes in Figure 7. All priority classes are assumed to have the same threshold, $\varphi_k = \varphi$ for all k. Further, the arrival rates are set to be equal for all four priority classes. The processing times are chosen constant for each class, but not equal. As can be read from the figure, $\lambda_k = 1/120$ for all k and $P_1 = 10$, $P_2 = 15$, $P_3 = 10$ and $P_4 = 15$. These values makes the figure comparable to the results obtained for a scheduling scenario simulated in Section 8. In Figure 7 expected flow-time is plotted against φ. The expected flow-times in strictly non-preemptive and preemptive systems are marked with dotted lines.

It should be noted from the figure that the jobs of highest priority are best off with a fully preemptive system. Jobs of lowest priority would prefer a strictly non-preemptive system when it comes to minimizing their expected flow-time. This is clear since class 1 jobs preempt jobs from all other priority classes and never get preempted them self, while the class 4 jobs on the other hand gain nothing from preemptions being allowed.

Considering this it is not very surprising that the other classes have optimal expected flow-times for thresholds between the two extrema 0 and 1. Class 2 jobs preempt class 3 and class 4 jobs, but are preempted by class 1 jobs. In the graph in Figure 7, $\varphi \approx 0.365$ is found to be the optimal threshold for class 2 jobs, as it minimizes the class 2 expected flow-time. A class 3 job can only preempt class 4 jobs. At the same time it may suffer preemptions due to both class 1 and class 2 jobs. Optimal threshold value for priority class 3 is consequently much closer to the fully preemptive

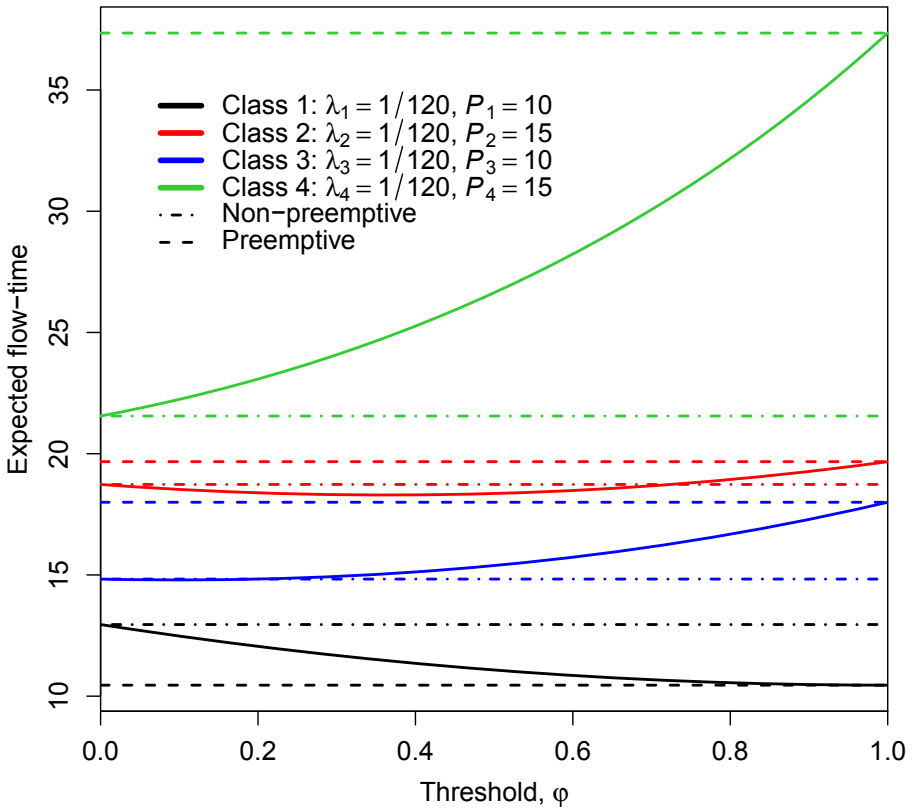


Figure 7: Expected flow-time in a combined priority system

system than that of priority class 2. In the graph $\varphi \approx 0.101$ is found to be the optimal threshold, with respect to minimization of the expected class 3 flow-time.

It should be mentioned that the expected flow-time of class 3 jobs is lower than the expected flow-time of class 2 jobs in this particular case. This is simply because the class 2 processing times are longer than the class 3 processing times.

Keep in mind that $\varphi_k = \varphi$ for all k in Figure 7. From another point of view, suppose that the threshold values are fixed, but not necessarily equal, for all priority classes but one. That is, φ_k is known and fixed for all $k \neq i$. An optimal threshold value for class i would always be $\varphi_i = 0$ as no preemption is better than some preemptions.

8 Scheduling on single track railway lines

It is now time to return to the scheduling problem regarding trains requested to operate on a single track railway line. First some assumptions allowing the problem to be viewed in the setting of queuing theory must be accounted for. It will be argued that the priority rule suggested by Jernbaneverket is, at certain important points, similar to the combined priority rule discussed in Section 7.

A simulated scheduling scenario will illustrate that the mean scheduled waiting times can be found by considering flow-times in a combined preemptive/non-preemptive priority system.

8.1 The scheduling problem

Briefly recapitulated from Section 1, scheduling is the task of assigning train paths to the train operators based on their requests. A train path is a segment of a railway line allocated to a single train in a given period of time.

The line segment in question is a single track block section that handles bidirectional train traffic. Due to safety regulations only one train is allowed to operate in a block section at a time. This to ensure a minimum spacing between trains and also because crossings (two trains passing each other in opposite directions) are impossible. Conflicts arise when paths preferred by the different train operators in some way overlap.

In the process of sorting the conflicts out trains are granted different priority on the line. The level of priority can be based on, for instance, train type or traveling direction. Operations are then authorized according to a chosen scheduling rule. It is convenient to recall the scheduling rule suggested by Jernbaneverket from Section 1.

> The train operators request to enter a block section at a given time. For a train from a class of low priority to be allowed to operate in the block section at the requested time, there must be enough time for it to traverse a predefined fraction φ of that block section before a train of higher priority is requested to operate. If the time criterion is not fulfilled, the train of low priority must wait. The train of high priority is then given authority to operate in the block section at the requested time.
>
> If the time criterion is fulfilled, it is the train of high priority that must wait while the train of low priority completes its operation. There might be several high priority requests placed for the time interval the line is allocated to the train of lower priority. Once the low priority train is scheduled to exit the block section, the train of highest priority, scheduled to wait, is granted authority

> to enter the section. In case of more than one train from the same priority class being scheduled to wait, the train with the earliest request is authorized to operate first. That is, a first-come-first-served principle is practiced within each priority class.

Adjusting the value of φ, $0 \leq \varphi \leq 1$, affect the effective capacity of the block section in question. A large φ will cause the block section to be unavailable to low priority trains and idle for a large amount of time. On the other hand, the high priority trains will be granted train paths close to their original requests.

Smaller values of φ will have the opposite effect; raising the overall effective capacity of the line. But also, small values of φ result in larger scheduled waiting times for trains of high priority.

It should be obvious by now that the scheduling problem is a priority queuing problem. However, a study of the scheduling problem in the light of the previously developed queuing theory calls for a few limiting assumptions.

8.2 Scheduling in light of queuing theory

The operators' requested times for entering the block section can be viewed as arrival times to a queuing system. Corresponding inter-arrival times are thus the time intervals between successive requested times. Since the different train operators operate independently of each other and their future wishes for train paths are unknown, it makes sense to model inter-arrival times as a stochastic process. Between requested times of entries to the block section, there are in fact observed many small time gaps and fewer large time gaps. To assume the intervals between arrivals being exponentially distributed thus seems appropriate (Handstanger, 2009, page 3). It follows immediately that the arrival process of interest is a Poisson processes.

Scheduling is done for a limited time period. A train requested to enter the block section in the scheduling period can be viewed as a job arriving at a queuing system. The train can be from one of K priority classes. The arrival rate λ_k, $k = 1, \ldots, K$, is the average number of trains from priority class k requested to enter the block section per time unit.

The block section only allows for one train to operate at a time. A train traversing the block section can be viewed as a job being processed. The queuing system of interest has only one processing unit and the processing time is the time a train needs to traverse the entire block section. These blocking times are assumed known and equal for all trains of a given type, as explained in Section 1. The processing times P_k are the deterministic class k blocking times, dependent only on the speed of the class k trains.

Assuming there is no limit on how many trains that can be requested to enter the block section within a given amount of time it should be clear that the scheduling problem can be described as an $M/D/1$ queuing system.

Here D denotes the degenerated, deterministic distribution of processing time. This is obviously a slightly specialized version of the $M/G/1$ queue with no variation in the class k processing times. It is worth mentioning that the queuing theory presented in Section 2–Section 7 is in fact valid also for stochastic processing times.

Now, for the suggested scheduling rule, scheduled waiting time plus the deterministic blocking time of a class k train equals the flow time of a class k job in a queuing system under the combined preemptive/non-preemptive priority rule. This despite a few obvious differences between the scheduling problem and the theoretical queuing system:

A train cannot be preempted from the block section in the same way as a job is said to be preempted in a theoretical queuing system. Instead it is simply held back and the block section kept empty. This is possible since all the requested times are known prior to the times for the authorized operations are set. It is as if whether or not a job is going to be preempted in the theoretical queuing system is known prior to the next arrivals.

The wasted processing times and breakdown times suffered by a job in the theoretical queuing system are part of the scheduled waiting time in the scheduling problem. The theoretical successful processing time is simply the deterministic blocking time. The time interval between a requested time for starting an operation and the actual scheduled time for exit from the block section equals the theoretical flow-time. It follows that the mean scheduled waiting time for a class k train is given by the expected flow-time in a combined non-preemptive/preemptive priority system, minus the actual class k processing time, P_k.

8.3 Simulation of a scheduling scenario

To visualize the similarity between the scheduling problem and the theoretical queuing system, a fictive scheduling scenario is considered.

A single track railway line connects a residential area and a harbor in the east and an industrial area in the west. Assume that both freight trains and passenger trains operate on this line. The prioritized task on this railway line is the transport of people from their homes to their work place. Westbound passenger trains thus form the prioritized class of trains. The second most important task is to transport raw materials from the harbor to the industrial area. Westbound freight trains form priority class 2.

The most flexible task is the transport of finished products from the industrial area to the harbor for shipping. It follows that eastbound freight trains are of lowest priority on the line, belonging to priority class 4. It is somewhat more important to transport people home from work after ended shifts, and the passenger trains traveling in direction east belong to priority class 3.

A block section on this line is of such length that a passenger train

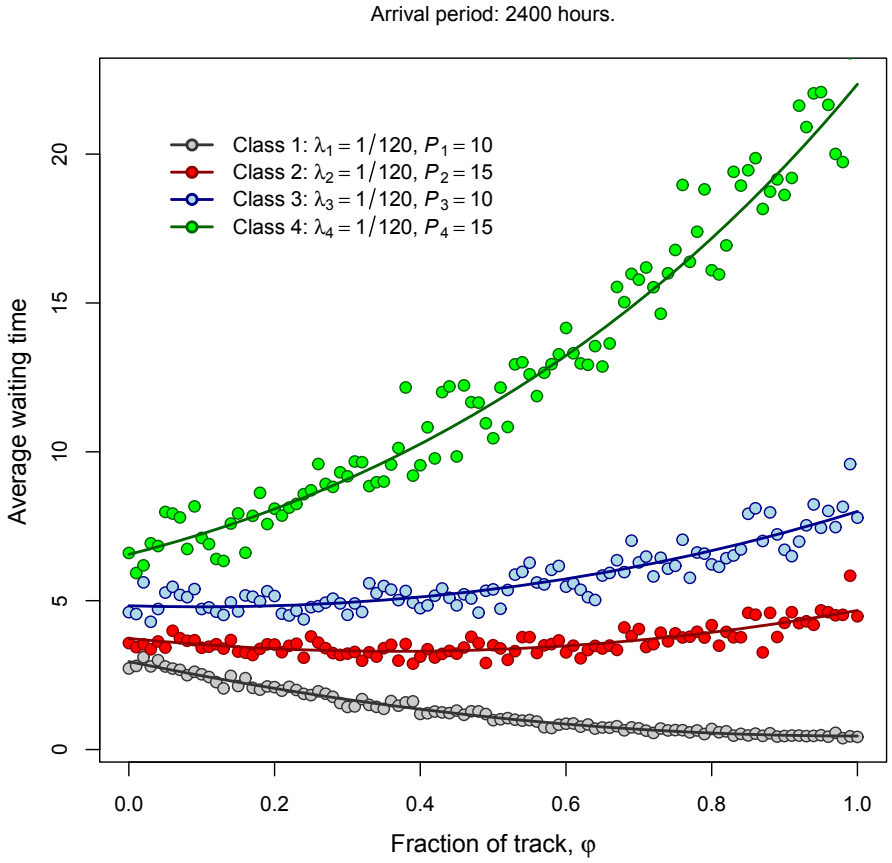


Figure 8: Average scheduled waiting time for trains requesting to traverse a single track block section. The length of the scheduling period is 100 days.

needs 10 minutes to traverse it. The somewhat slower freight trains need 15 minutes to traverse the same section. Now, if there on average is a request for operating one train from each class every second hour, the priority classes have the following properties:

- Priority class 1: Westbound passenger trains. $\lambda_1 = 1/120$, $P_1 = 10$.
- Priority class 2: Westbound freight trains. $\lambda_2 = 1/120$, $P_2 = 15$.
- Priority class 3: Eastbound passenger trains. $\lambda_3 = 1/120$, $P_3 = 10$.
- Priority class 4: Eastbound freight trains. $\lambda_4 = 1/120$, $P_4 = 15$.

The situation is simulated using the source code in Appendix C (R Development Core Team, 2010). Scheduling is assumed to be done for a period

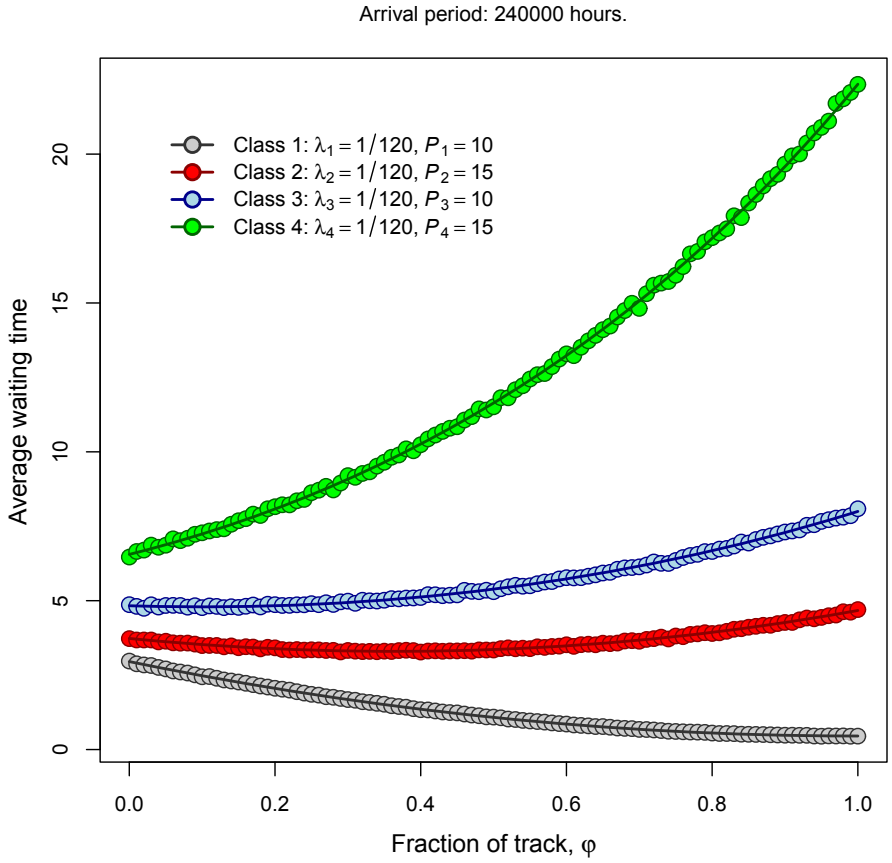


Figure 9: Average scheduled waiting time for trains requesting to traverse a single track block section. The length of the scheduling period is 10000 days.

of 100 days, or 2400 hours. Following the suggested priority rule strictly, leads to average scheduled waiting times plotted in Figure 8. The simulations are done repeatedly for 100 different φ-values in the interval from 0 to 1. The threshold value φ is assumed to be the same for all the priority classes in each simulation. The solid lines represent the steady state solution $\mathrm{E}[F_k] - P_k$ as a function of φ, where $\mathrm{E}[F_k]$ is given by Equation (43) in Section 7.

As expected, from the theoretical results from Section 7, the high priority trains are allowed to operate closer to their requested times when φ is large. Consequently, the low priority trains suffer large scheduled waiting times.

From Figure 8 it is clear that the simulated average scheduled waiting times vary around the theoretical steady state solution. Using the same

priority classes as above, requested times are simulated for a period of 10000 days – almost 30 years. The result is shown in Figure 9. Still there are small deviations in average scheduled waiting times from the theoretical, expected values. This suggest that there is actually quite a large amount of variation in the suffered scheduled waiting times within each priority class.

As was the objective of this project, a theoretical model describing the scheduling problem has been obtained. The first moment otained for class k flow-time gives an accurate description of the mean, class k, scheduled waiting time. To investigate the variation observed in the simulations closer, the second moment of flow-time must be obtained. This can be done by differentiating the Laplace transform associated with flow-time once more. Though simple in theory, in practice this is a lengthy operation beyond the time frame of this study.

9 Conclusion

Prior to the initiation of this work the Norwegian railway owner and operator, Jernbaneverket, proposed a priority based decision rule to be used during the process of scheduling trains operating on single track railway lines. It was shown that this suggested scheduling rule, under certain limiting assumptions, could be studied in the setting of queuing theory.

The main part of this study has been the development and analysis of a threshold based, combined preemptive/non-preemptive priority discipline. Under this discipline, preemptions of low prioritized jobs were allowed during the early stage of processing only. The effect of the combined priority rule on the flow-times in an $M/G/1$ queuing system with an arbitrary number of priority classes was studied in detail. An exact solution for expected flow-times in a system in steady state was obtained under the assumptions of known arrival rates, mean processing times and thresholds for each priority class.

The priority based scheduling rule for trains operating on a segment of a single track railway line was modeled by the theoretical, combined priority rule. The focus has been on scheduled waiting times, which is the time between the requested time for an operation and the actual, authorized time for that operation. The theoretical expression for expected flow-times obtained from queuing theory was, when adjusted down by the actual, nonrandom processing time, seen to yield an accurate expression for average scheduled waiting times. It was the main objective of this study to obtain a theoretical model suitable to describe the scheduling problem – and the scheduled waiting times in particular – under the proposed, priority based decision rule. That objective was successfully achieved, as theory fitted simulated data nicely.

For a simulated scheduling scenario there was observed variation in the

scheduled waiting times. Trains suffering large scheduled waiting times represent a much bigger problem to the railway operator than trains authorized to operate at the requested times. When deciding on how to adjust the threshold values to optimize the usage of the railway line in question, it is consequently not only the mean scheduled waiting time that is of importance. It may also be vital to avoid the occasionally huge scheduled waiting times suffered by a few trains. A natural extension of the present work would therefore be an investigation of the flow-time variance. This calls for the second moments of flow-time to be obtained. While it is a straightforward procedure in theory it is a tedious task in practice. Some of the most time consuming work may be avoided by use of symbolic mathematics software packages, such as Maple or Mathematica. Further studies of simulations may also provide useful insight.

Another aspect worth considering is that train scheduling in fact allows for negative waiting times. In some cases it is possible to schedule operations earlier than the actual request in order to avoid conflicts. It is likely that conventional queuing theory needs a considerable amount of modification to handle that scenario.

References

Y. Z. Cho and C. K. Un. Analysis of the $M/G/1$ queue under a combined preemptive/nonpreemptive priority discipline. *IEEE Transactions on Communications*, 41(1):132–141, 1993.

Richard W. Conway, William L. Maxwell, and Louis W. Miller. *Theory of Scheduling*. Addison-Wesley Publishing Company, 1967.

Steve Drekic and David A. Stanford. Threshold-based interventions to optimize performance in preemptive priority queues. *Queueing Systems*, 35 (1-4):289–315, 2000.

Johan Narvestad Fatnes. Scheduled waiting time on a single track line under a predefined priority rule. Project thesis, TMA4500, 2009.

Anne Christine Torp Handstanger. *Scheduled waiting time from crossing on single track railway lines*. PhD thesis, The Norwegian University of Science and Thechnology, 2009.

Ingo Arne Hansen and Jorn Pachl. *Railway Timetable and Traffic*. Eurail Press, 2008.

N. K. Jaiswal. *Priority queues*. New York, Academic Press, 1968.

Leonard Kleinrock. *Queueing systems, Volume I: Theory*. John Wiley & Sons, Inc., 1975.

Leonard Kleinrock. *Queueing systems, Volume II: Computer Applications.* John Wiley & Sons, Inc., 1976.

Leonard Kleinrock and Richard Gail. *Queueing Systems: Problems and Solutions.* John Wiley & Sons, Inc., 1996.

Erwin Kreyzig. *Advanced Engineering Mathematics.* John Wiley & Sons, Inc., 8th edition, 1999.

John D. C. Little. A Proof for the Queueing Formula: $L = \lambda W$. *Operations Research*, 9(3):383–387, 1961.

R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2010. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

Sheldon M. Ross. *Introduction to probability models.* Academic Press, 9th edition, 2007.

Svein Skartsæterhagen. Kapasitet på jernbanestrekninger. Prepared for NSB banedivisjonen, Teknisk kontor, Institutt for energiteknikk, Norway, 1993.

Hideaki Takagi. *Queueing analysis. A foundation of performance evaluation, Volume 1: Vacation and Priority Systems, Part 1.* John Wiley & Sons, Inc., 1991.

Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, and Keying Ye. *Probability and statistics for engingneers and scientists.* Pearson Prentice Hall, 8th edition, 2007.

Ronald W. Wolff. Work-conserving priorities. *Journal of Applied Probability*, 7(2):327–337, 1970.

A Transform theory

This appendix is a short introduction to the Laplace transform and some of its properties. In queuing theory the Laplace transforms of probability density functions are of importance mainly because of their close connection to the moment generating functions of random variables. They provide an alternative characterization of random variables and a way to calculate their moments. It is convenient to consider the Laplace transform rather than the moment generating function itself, as the Laplace transform always is between 0 and 1 when the random variable is non-negative.

The material is based on Kreyzig (1999, Chapter 5) and the notation is chosen to fit the main part of this text.

A.1 The Laplace transform

Consider a function $g(t)$ continuous for all $t \geq 0$. The Laplace transform of $g(t)$ is defined as

$$\mathcal{L}\{g(t)\} = \gamma(z) = \int_{t=0}^{\infty} e^{-zt} g(t) \mathrm{d}t. \tag{44}$$

If $g(t)$ is a piecewise continuous function on every finite interval in the range $t \geq 0$ and satisfies

$$|g(t)| \leq M e^{kt} \tag{45}$$

for all $t \geq 0$ and some constants M and k, then the Laplace transform of $g(t)$ exists for all $z > k$. Furthermore, if the Laplace transform of a given function exists, it is uniquely determined. More details and a proof of the existence theorem can be found in Kreyzig (1999, page 256). Several important properties of the Laplace transform follow.

First, the Laplace transform is a linear transform. That is, for functions $g_1(t)$ and $g_2(t)$ of continuous time and for any constants a and b a straight forward calculation yields

$$\begin{aligned}
\mathcal{L}\{ag_1(t) + bg_2(t)\} &= \int_{t=0}^{\infty} e^{-zt} \left(ag_1(t) + bg_2(t)\right) \mathrm{d}t \\
&= a \int_{t=0}^{\infty} e^{-zt} g_1(t) \mathrm{d}t + b \int_{t=0}^{\infty} e^{-zt} g_2(t) \mathrm{d}t \\
&= a\gamma_1(z) + b\gamma_2(z).
\end{aligned}$$

Another important property is that of the first shifting theorem. If $g(t)$ has the Laplace transform $\gamma(z)$, then $e^{-at}g(t)$ has the Laplace transform $\gamma(z-a)$. This is easily obtained from

$$\gamma(z-a) = \int_{t=0}^{\infty} e^{-(z-a)t} g(t) \mathrm{d}t = \int_{t=0}^{\infty} e^{-zt} \left[e^{at} g(t)\right] \mathrm{d}t = \mathcal{L}\left\{e^{at} g(t)\right\}.$$

From time to time it proves useful to recognize the Laplace transform of the derivative of $g(t)$. Consider $g'(t)$ continuous[3] for all $t \geq 0$. By definition and integration by parts

$$\begin{aligned}\mathcal{L}\left\{g'(t)\right\} &= \int_{t=0}^{\infty} e^{-zt} g'(t) \mathrm{d}t \\ &= \left[e^{-zt} g(t)\right]_0^{\infty} + z \int_{t=0}^{\infty} e^{-zt} g(t) \mathrm{d}t \\ &= z\gamma(z) - g(0).\end{aligned}$$

More generally, the expression for the Laplace transform of the ith derivative of $g(t)$ is

$$\mathcal{L}\left\{g^{(i)}(t)\right\} = z^i \gamma(z) - z^{i-1} g(0) - z^{i-2} g'(0) - \cdots - g^{(i-1)}(0).$$

which can be proven by induction (Kreyzig, 1999, page 259).

As the derivatives of $g(t)$ can be obtained essentially by multiplying the transform $\gamma(z)$ by z, integration can be done by dividing on z. That is, for $g(t)$ piecewise continuous, satisfying (45) for some k and M

$$\mathcal{L}\left\{\int_{\tau=0}^{t} g(\tau)\mathrm{d}\tau\right\} = \frac{1}{z}\gamma(z). \tag{46}$$

Once again the reader is referred to Kreyzig (1999, page 262) for details.

To move on, the unit step function and Dirac's Delta function, also known as the unit impulse function, must be defined. The unit step function $u(t-a)$ is 0 for $t < a$ by definition, has a jump of size 1 at $t = a$ (where it can be left undefined) and is 1 for $t > a$. That is,

$$u(t-a) = \begin{cases} 0 & \text{if } t < a \\ 1 & \text{if } t > a \end{cases}.$$

The unit impulse function is important for handling discontinuities and their derivatives. The impulse of a force $g(t)$ over time interval $a \leq t \leq a+k$ is defined to be the integral of $g(t)$ from a to $a+k$. Of particular interest is the case where the force acts only for an instance. To deal with that situation, consider the function

$$g_k(t-a) = \begin{cases} 1/k & \text{if } a \leq t \leq a+k \\ 0 & \text{otherwise} \end{cases}, \qquad a \geq 0.$$

The impulse I_k is 1 since

$$I_k = \int_{t=0}^{\infty} g_k(t-a)\mathrm{d}t = \int_{t=a}^{a+k} \frac{1}{k}\mathrm{d}t = 1.$$

[3]For a piecewise continuous $g'(t)$ the result is similar, but the integration must be broken up into parts such that $g'(t)$ is continuous on each part.

The unit impulse function[4] is defined as

$$\delta(t-a) = \lim_{k\to\infty} g_k(t-a).$$

The unit step function and the unit impulse function has Laplace transformations given as

$$\mathcal{L}\{u(t-a)\} = \frac{e^{-az}}{z} \tag{47}$$

and

$$\mathcal{L}\{\delta(t-a)\} = e^{-az}$$

respectively.

By using the unit step function it is possible to establish another important relationship described by the second shifting theorem. If $g(t)$ has the transform $\gamma(z)$, the shifted function

$$g(t-a)u(t-a) = \begin{cases} 0, & t < a \\ g(t-a), & t > a \end{cases}$$

has a Laplace transform given as

$$\mathcal{L}\{g(t-a)u(t-a)\} = e^{-as}\gamma(s)$$

(Kreyzig, 1999, page 267).

The final property of the Laplace transform discussed in this extremely brief coverage of the topic is the convolution property. It has to do with the products of transforms. For two functions, $f(t)$ and $g(t)$, that satisfy inequality (45), the product $\eta(z)$ of their transforms, $\phi(z)$ and $\gamma(z)$, is the transform of the convolution of $f(t)$ and $g(t)$, defined as

$$(f * g)(t) = \int_{\tau=0}^{t} f(\tau)g(t-\tau)\mathrm{d}\tau. \tag{48}$$

That is, $\mathcal{L}\{(f*g)(t)\} = \phi(z)\gamma(z)$. For random variables X and Y with probability density functions $f(t)$ and $g(t)$, respectively, the convolution $(f*g)(t)$ gives the density of $X+Y$. For further details, see Kreyzig (1999, page 279–281).

A.2 Moment generating functions

The ith moment about the origin of a continuous random variable X is $\mathrm{E}\left[X^i\right]$, that is

$$\mathrm{E}\left[X^i\right] = \int_{-\infty}^{\infty} x^i f(x)\mathrm{d}x$$

[4]Not a function in the ordinary sense.

(Walpole et al., 2007, page 220) where $f(x)$ is the probability density function of X. The first moment of the random variable X is its mean. Furthermore, the variance of X can be written in terms of the first and second moment as $\mathrm{Var}[X] = \mathrm{E}[X^2] - (\mathrm{E}[X])^2$.

The moment generating function of the continuous random variable X is given by $M_X(v) = \mathrm{E}\left[e^{vX}\right]$, that is

$$M_X(v) = \mathrm{E}\left[e^{vX}\right] = \int_{-\infty}^{\infty} e^{vx} f(x) \mathrm{d}x. \tag{49}$$

Existence of the moment generating function depends entirely on the convergence of the integral in Equation (49).

The obvious purpose of the moment generating functions is to establish moments of random variables. These can be obtained from the relationship

$$\left.\frac{\mathrm{d}^i M_X(v)}{\mathrm{d}v^i}\right|_{v=0} = \mathrm{E}\left[X^i\right],$$

as shown in Walpole et al. (2007).

When one compares the definition of the Laplace transform, given by Equation (44), to the definition of moment generating functions, given by Equation (49), one realizes that the two are quite similar. In fact, for a positive, random variable P, with cumulative distribution $G(p) = \Pr(P \leq p)$, the corresponding Laplace transform is

$$\gamma(z) = \mathrm{E}\left[e^{-zP}\right] = \int_{p=0}^{\infty} e^{-zp} g(p) \mathrm{d}p.$$

Hence, moments of P can be found from the formula

$$\lim_{z \to 0} \frac{\mathrm{d}^i \gamma(z)}{\mathrm{d}z^i} = (-1)^i \mathrm{E}\left[P^i\right]. \tag{50}$$

This relationship makes it possible to obtain important properties of a distribution from the corresponding Laplace transform.

Finally it is convenient to note that as long as $g(x)$ is a probability density function

$$\gamma(z) = \int_{-\infty}^{\infty} e^{-zp} g(p) \mathrm{d}p \leq 1,$$

and $\gamma(z) = 1$ only if $z = 0$.

B Renewal theory

The renewal process is a generalization of the Poisson process. It is a counting process $\{N(t),\, t \geq 0\}$ for which the time $\{P_1, P_2, \dots\}$ between successive

events are independently and identically distributed according to an arbitrary distribution (Ross, 2007, page 417). Each event is called a renewal and P_i, $i = \{1, 2, \dots\}$ are called renewal intervals. This section deals with two, at least in the setting of queuing theory, very important results from the renewal theory.

Both results are briefly presented both by Conway et al. (1967, page 146–147) and by Kleinrock (1975, page 169–174). A more complete treatment of the renewal theory is given by Ross (2007, Chapter 7).

B.1 The inspection paradox

In the setting of queuing theory the problem in question is this: Given a job being processed, what is the distribution of the remaining processing time for this job when another job arrives at some random point in time?

Assume that job i goes on the machine for processing at time τ_i, and that the processing of the first job begins at time 0. The renewal process $\{N(t), t \geq 0\}$ is in this case the number of jobs that have entered the machine for processing at time t. A renewal event occurs when one job departs from the system. Idle periods are of no interest at this point, so it is assumed that there always is a new job waiting for processing. Hence, when the processing of job i is completed, the processing of job $i + 1$ starts immediately. The intervals $P_i = \tau_{i+1} - \tau_i$ is thus the processing time of job i. Further, it is assumed that the processing intervals P_i, $k = 1, 2, \dots$, are independently and identically distributed random variables distributed according to

$$G(p) = \Pr(P \leq p).$$

At some random point in time, say t, a job arrives at the queuing system. Assume that $N(t) = n$, job n is being processed, and denote by $Y(t)$ the remaining processing time of that particular job at time t.

In the terminology of renewal theory $\{\tau_i\}$ form a sequence of renewal points in time. $Y(t)$ is the residual life of the selected job n. $X = \tau_{n+1} - \tau_n$ is the lifetime[5] of the particular job n and $A(t) = X - Y(t)$ is the age of job n at time t. The cumulative distributions of residual life, Y, and the selected lifetime X are $F_Y(y) = \Pr(Y \leq y)$ and $F_X(x) = \Pr(X \leq x)$, respectively.

The results of this section are based on the observation that long intervals between renewal points occupy larger segments of the time axis than short intervals[6] Thus, a long interval is more likely to cover the randomly selected point t than a short interval. It follows that the lifetime of the selected job n, X, is *not* distributed according to $G(p)$. This is commonly referred to as the inspection paradox (Ross, 2007, page 455).

[5]Note that $X = P_n$, but the choice of X simplifies the notation as the need for subscripts are avoided.

[6]This observation must somehow be accepted. However, the main result of this section is derived in more rigorous manners in Kleinrock and Gail (1996, page 120–125).

For the continuous variable X, the relative occurrence of x is given by

$$\Pr(x < X \leq x + \mathrm{d}x) = f_X(x)\mathrm{d}x,$$

for a infinitesimal $\mathrm{d}x$.

The probability of an event – in this case the arrival of a job at the queuing system – occurring in a processing interval of length x should be proportional to that length and to the relative occurrence of such intervals. The latter is given by $g(x)\mathrm{d}x$. Thus, the probability that the selected interval X is of length x is

$$\Pr(x < X \leq x + \mathrm{d}x) = Kxg(x)\mathrm{d}x,$$

where K must be evaluated to properly normalize this density. This is done by integrating both sides of the previous equation from zero to infinity:

$$\int_{x=0}^{\infty} f_X(x)\mathrm{d}x = K\int_{x=0}^{\infty} xg(x)\mathrm{d}x.$$

Directly from this K is given by $K = (\mathrm{E}[P])^{-1}$. The probability density function of the selected processing interval X can from this bewritten as

$$f_X(x) = \frac{xg(x)}{\mathrm{E}[P]} \tag{51}$$

Now, if the length of the selected processing interval X is known to be x, the probability that the residual life $Y(t)$ does not exceed y is

$$\Pr(Y(t) \leq y \mid X = x) = \frac{y}{x}, \quad \text{for } 0 \leq y \leq x,$$

since $Y(t)$ depends only on a randomly chosen, and thus uniformly distributed, point t in the given interval of length x.

Using the uniformity of the conditional distribution above, the joint density of X and $Y(t)$ follows from Bayes' rule (Walpole et al., 2007, page 71) and Equation (51) as

$$\begin{aligned}
&\Pr(y < Y(t) \leq y + \mathrm{d}y,\, x < X \leq x + \mathrm{d}x) \\
&= \Pr(y < Y(t) \leq y + \mathrm{d}y \mid X = x)\Pr(x < X \leq x + \mathrm{d}x) \\
&= \Pr(Y(t) \leq \mathrm{d}y \mid X = x)\Pr(x < X \leq x + \mathrm{d}x) \\
&= \left(\frac{\mathrm{d}y}{x}\right)\left(\frac{xg(x)}{\mathrm{E}[P]}\mathrm{d}x\right) \\
&= \frac{g(x)\mathrm{d}y\mathrm{d}x}{\mathrm{E}[P]} \quad \text{for } 0 \leq y \leq x.
\end{aligned} \tag{52}$$

To obtain the marginal distribution of $Y(t)$ the joint probability density function of $Y(t)$ and X is integrated with respect to x:

$$\begin{aligned} f_Y(y)\mathrm{d}y &= \frac{\mathrm{d}y}{\mathrm{E}\left[P\right]} \int_{x=y}^{\infty} g(x)\mathrm{d}x \\ &= \frac{\mathrm{d}y}{\mathrm{E}\left[P\right]} \left[1 - \int_{x=0}^{y} g(x)\mathrm{d}x\right] \\ &= \frac{\mathrm{d}y}{\mathrm{E}\left[P\right]} \left[1 - G(y)\right]. \end{aligned}$$

Finally, this gives the distribution of residual life $Y(t)$ in terms of the distribution of the processing times and the mean processing time as[7]

$$f_Y(y) = \frac{1 - G(y)}{\mathrm{E}\left[P\right]}. \tag{53}$$

From Equation (46), Equation (47) and the linearity property of the Laplace transform, all given in Appendix A, the Laplace transform $\phi_Y(z)$ associated with $f_Y(y)$ can be obtained as

$$\begin{aligned} \phi_Y(z) &= \frac{1}{\mathrm{E}\left[P\right]} \left[\mathcal{L}\left\{1\right\} - \mathcal{L}\left\{\int_{t=0}^{y} g(t)\mathrm{d}t\right\}\right] \\ &= \frac{1}{\mathrm{E}\left[P\right]} \left[\frac{1}{z} - \frac{\gamma(z)}{z}\right] \\ &= \frac{1 - \gamma(z)}{z\mathrm{E}\left[P\right]}, \end{aligned} \tag{54}$$

where $\gamma(z)$ is the Laplace transform associated with the probability density function $g(x)$.

The ith moment of the processing time is $\mathrm{E}\left[P^i\right]$. The ith moment of residual life is $\mathrm{E}\left[Y^i\right]$ and can be obtained from $\phi_Y(z)$ by using Equation (50) from Appendix A. Derivation of $\phi_Y(z)$ with respect to z yields

$$\frac{\mathrm{d}\psi_Y(z)}{\mathrm{d}z} = \frac{\gamma(z) - z\gamma'(z) - 1}{z^2\mathrm{E}\left[P\right]}.$$

By Equation (44), taking the limit of this as z goes to zero results in a 0/0-expression as $\gamma(0) = 1$. Application of l'Hôpital's rule yields

$$\lim_{z\to 0} \frac{\mathrm{d}\phi_Y(z)}{\mathrm{d}z} = -\frac{\gamma''(0)}{2\mathrm{E}\left[P\right]}.$$

Here, as in the rest of the text, the above notation is defined by

$$\gamma^{(i)}(0) = \lim_{z\to 0} \frac{\mathrm{d}^i\gamma(z)}{\mathrm{d}z^i}.$$

[7] It can be shown that the limiting probability density function for age $A(t)$ is the same as for residual life $Y(t)$.

Now, since $\gamma''(0) = \mathrm{E}\left[P^2\right]$,

$$\mathrm{E}\left[Y\right] = (-1)\phi_Y''(0) = \frac{\mathrm{E}\left[P^2\right]}{2\mathrm{E}\left[P\right]}.$$

Written in terms of the variance of the processing times, $\mathrm{Var}\left[P\right] = \mathrm{E}\left[P^2\right] - (\mathrm{E}\left[P\right])^2$, the first moment of residual life Y becomes

$$\mathrm{E}\left[Y\right] = \frac{\mathrm{E}\left[P_k\right]}{2} + \frac{\mathrm{Var}\left[P_k\right]}{2\mathrm{E}\left[P_k\right]}.$$

From this $\mathrm{E}\left[Y\right] \geq \frac{1}{2}\mathrm{E}\left[P\right]$ and growing with $\mathrm{Var}\left[P\right]$.

In case of the processing times being exponentially distributed with rate $1/\lambda$, the expected residual life of a selected processing interval is $\mathrm{E}\left[Y\right] = 1/\lambda = \mathrm{E}\left[P\right]$. That is, when a Poisson process is randomly intercepted, one expect to observe $2/\lambda$ time between two successive processing completions, even though the expected time between two completions is $1/\lambda$ for that process.

B.2 The elementary renewal theorem

The result of the elementary renewal theorem is important in the discussion of steady state probabilities.

Let $N(t)$ be the number of renewals in an interval of length t. The expected number of renewals in an interval of length t is called the renewal function and denoted by $\mathrm{E}\left[N(t)\right]$. Properties of this function are discussed by Ross (2007, page 419–423). The essence of the elementary renewal theorem is

$$\lim_{t\to\infty} \frac{\mathrm{E}\left[N(t)\right]}{t} = \frac{1}{\mathrm{E}\left[P\right]} \tag{55}$$

(Ross, 2007, page 425). Here $\mathrm{E}\left[P\right]$ is the expected length of a renewal interval and thus the expected time between renewals. $r = 1/\mathrm{E}\left[P\right]$ is called the rate of the renewal process. It is quite intuitive that the average rate at which renewals occur is 1 per every $\mathrm{E}\left[P\right]$ time units.

C Source code

This appendix contain the source code used for simulations of the train problem. The code is written in `R` (R Development Core Team, 2010), a free software environment for statistical computing and graphics.

The function `Queue` is used to calculate waiting time for a given set of arriving trains. The advantage of the function is that it can handle any form for requests and does not depend on the arrival process being Markovian.

```
Queue <- function(arrival, process, priority, phi) {
  # Computes flow-time and waiting time for a given set of job arrivals to a
  # combined preemptive/non-preemptive priority queue.
  #
  # Args:
  #   arrival:  vector of sorted arrival times in absolute time
  #   process:  vector of processing times associated with the arrival times
  #             given in arrival.
  #   priority: vector of integers (1,...,N) giving the priority of the jobs
  #             arriving according to the arrival times given in arrival.
  #   phi:      a vector of the porportion of processing time the different
  #             classes must complete before the system becomes non-preemptive.
  #             phi[k] is the treshold associated with priority class k.
  #
  # Returns:
  #   A list containg vectors of
  #      * corresponding priorities
  #      * flow-time
  #      * waiting time
  #   The vectors correspond to the entries of the given arrival and process.
    # Error handling
  if (length(arrival) != length(process)) {
    stop("Arguments arrival and process have invalid lengths: ",
         length(arrival), " and ", length(process), ".");
  }
    # Initiating the system
  pri.classes <- seq(1:max(priority));
  machine.available <- 0;
  starttimes <- c();
  stoptimes <- c();
  i <- rep(0,max(priority));  # Position in sub-arrival time vector.
  n <- c();  # Number of jobs in the different priority classes.
  first.arrival <- c();         # Arrival time of first class k job to be processed.
  first.process <- c();         # Processing time of first class k job to be processed.
    # Set up subset of arrivals and processingtimes
  subset.arrival <- list(NULL);
  subset.process <- list(NULL);
  for (l in pri.classes) {
    subset.arrival[[l]] <- subset(arrival,priority==l);
    subset.process[[l]] <- subset(process,priority==l);
  }
  for (l in pri.classes) {
    n <- c(n,length(subset.arrival[[l]]));
    first.arrival <- c(first.arrival, subset.arrival[[l]][i[l]+1]);
    first.process <- c(first.process, subset.process[[l]][i[l]+1]);
  }
    # Remove priority classes not present
  remaining.classes <- pri.classes[which(!n==0)];
  remaining.phi <- phi[which(!n==0)];
  first.arrival <- first.arrival[which(!n==0)];
  first.process <- first.process[which(!n==0)];
    # Calculate when job reaches their critical point if starting as soon as
    # possible. The job with lowest critical.point value is processed first.
  critical.point <- first.arrival + remaining.phi*first.process;
  if (length(remaining.classes) > 1) {
    for (j in length(remaining.classes):2) {
      if (critical.point[j] > min(first.arrival[1:(j-1)])) {
        critical.point[j] <- Inf;
      }
    }
  }
  class <- remaining.classes[which(critical.point
```

```
                                ==min(critical.point))];
global.index <- which(priority==class)[i[class] + 1];
starttimes[global.index] <- arrival[global.index];
machine.available <- arrival[global.index] + process[global.index];
stoptimes[global.index] <- machine.available;
i[class] <- i[class]+1;  # Update class pointer
while (!all(i==n)) {
    # Remove classes finished completely
  keep <- c();
  for (j in 1:length(i)) {
    if (i[j]!=n[j]) {
      keep <- c(keep,j);
    }
  }
  remaining.classes <- pri.classes[keep];
  remaining.phi <- phi[keep];
    # Find next job to be processed, and update starttimes/stoptimes
  first.arrival <- c();
  first.process <- c();
  for (l in remaining.classes) {
    first.arrival <- c(first.arrival, subset.arrival[[l]][i[l]+1]);
    first.process <- c(first.process, subset.process[[l]][i[l]+1]);
  }
  critical.point <- c();
  begin.process <- c()
  for (l in 1:length(remaining.classes)) {
    if (first.arrival[l] < machine.available) {
      critical.point <- c(critical.point, machine.available
                          + remaining.phi[l]*first.process[l]);
      begin.process <- c(begin.process, machine.available);
    }
    else {
      critical.point <- c(critical.point, first.arrival[l]
                          + remaining.phi[l]*first.process[l]);
      begin.process <- c(begin.process, first.arrival[l]);
    }
  }
  if (length(remaining.classes) > 1) {
    for (j in length(remaining.classes):2) {
      if (critical.point[j] > min(first.arrival[1:(j-1)])) {
        critical.point[j] <- Inf;
      }
    }
  }
  local.index <- which(critical.point==min(critical.point))[1];
  class <- remaining.classes[local.index];
  global.index <- which(priority==class)[i[class] + 1];
  starttimes[global.index] <- begin.process[local.index];
  machine.available <- begin.process[local.index] + process[global.index];
  stoptimes[global.index] <- machine.available;
  i[class] <- i[class] + 1;  # Update class pointer
}
  # Calculate flow-time and waiting time
flow.time <- stoptimes - arrival;
waiting.time <- starttimes - arrival;
  # Construct list with results to return
result <- list(priority=priority,
               stop=stoptimes,
               start=starttimes,
               flow=flow.time,
               waiting=waiting.time);
return(result);
```

```
}
```

The function in `GenerateQueue` was used to generate train arrivals with exponential inter-arrival time.

```
GenerateQueue <- function(rate, mean.process, kPeriod) {
  # Generates input vectors to the function Queue() under the assumption of
  # deterministic processing times and exponentially distributed inter-arrival
  # times for each priority class. Note: phi[1] should always be 0.
  #
  # Args:
  #  rate:          Vector of arrival rates corresponding to the priority
  #                 classes
  #  mean.process:  Vector of mean processing time of corresponding priority
  #                 class.
  #  kPeriod:       Constant, giving the period jobs arrive at the system
  #
  # Returns:
  #  A list with sorted arrival times and corresponding priority and process
  #  vector containing priority and processing time of each arriving
  #  job, respectively.
  total.int.arrival <- c();
  priority <- c();
  process <- c();
  arrival.rate <- c();
    for (i in 1:length(rate)) {
      inter.arrival <- rexp(1,rate[i]);  # sub inter-arrival time
      while (sum(inter.arrival) < kPeriod) {
        new.int.arrival <- rexp(1,rate[i]);  # draw a new inter-arrival time
        if (sum(inter.arrival) + new.int.arrival < kPeriod) {
          inter.arrival <- c(inter.arrival, new.int.arrival);
        }
        else {
          break;
        }
      }
      if (sum(inter.arrival) > kPeriod) {
        inter.arrival <- c();
      }
      absolute.arrival <- cumsum(inter.arrival);
      total.int.arrival <- c(total.int.arrival, absolute.arrival);
      priority <- c(priority, rep(i, length(absolute.arrival)));
      process <- c(process, rep(mean.process[i], length(absolute.arrival)))
      arrival.rate <- c(arrival.rate, rep(rate[i], length(absolute.arrival)))
    }
  priority <- priority[order(total.int.arrival)];
  process <- process[order(total.int.arrival)];
  arrival.rate <- arrival.rate[order(total.int.arrival)];
  arrival <- sort(total.int.arrival);
  result <- list(arrival=arrival,
                 rate=arrival.rate,
                 priority=priority,
                 process=process);
  return(result);
}
```