Analyse en optimalisatie van discrete-tijd-wachtlijnmodellen met generalized-processor-sharing

Analysis and Optimization of Discrete-Time Generalized Processor Sharing Queues

Jasper Vanlerberghe

Promotoren: prof. dr. ir. J. Walraevens, prof. dr. ir. H. Bruneel
Proefschrift ingediend tot het behalen van de graad van
Doctor in de ingenieurswetenschappen

Vakgroep Telecommunicatie en Informatieverwerking
Voorzitter: prof. dr. ir. J. Walraevens
Faculteit Ingenieurswetenschappen en Architectuur
Academiejaar 2018 - 2019

UNIVERSITEIT
GENT

ISBN 978-94-6355-162-5
NUR 919, 986
Wettelijk depot: D/2018/10.500/80

Dankwoord

De laatsten zullen de eersten zijn. Eindelijk ben ik aan het laatste — en meest uitgestelde — hoofdstuk van dit boek begonnen, wat grappig genoeg het eerste is dat jullie zullen lezen. Het is een dankwoord, want hoewel ik als enige auteur van dit boek genoteerd sta, was het er zonder de expliciete en impliciete hulp van velen simpelweg nooit gekomen.

Na een e-mail van professor Herwig Bruneel had ik 25 januari 2012, een woensdagnamiddag, met hem afgesproken. We zouden het in het algemeen hebben over de modaliteiten van een doctoraat. Ik wist immers op dat moment nog niet hoe het onderzoek voor een thesis mij zou bevallen, laat staan of ik een eventueel doctoraat dan wel in de wachtlijntheorie wilde doen. Anderhalf uur later stapte ik terug buiten en wist ik niet goed wat er allemaal gebeurd was. We hadden het inderdaad kort over die modaliteiten van een doctoraat gehad maar waren het grootste gedeelte van de tijd afgedwaald. Onder de indruk van het aangename gesprek, had ik er meteen spijt van dat ik nooit in zijn theorieles aanwezig was geweest. Herwig, ik weet niet of ik je dat ooit verteld heb, maar bij deze 'sorry'[1]. De rest is geschiedenis. Mijn eerste woord van dank gaat dan ook uit naar mijn promotor Herwig. Zonder jou was ik nooit bij SMACS/TELIN terecht gekomen. Ik wens je bovendien graag uitdrukkelijk te bedanken voor de ontspannen sfeer die je vorm hebt gegeven in de vakgroep en het feit dat ik me op het einde van de maand nooit zorgen hoefde te maken of ik wel betaald zou worden.

Van bij mijn begin bij SMACS nam professor Joris Walraevens mij op sleeptouw. Alle resultaten in dit boek bouwen voort op zijn werk en de ideeën zijn ontsproten tijdens vruchtbare discussies met hem. Hij heeft me alle kneepjes van het vak geleerd en hij heeft bladen vol teksten tijdig nagelezen, niet in het minst de laatste weken voor het indienen van dit boek. Joris, ik bewonder vooral je opendeurenbeleid en je toegankelijkheid. Zelden maakte ik vooraf een afspraak, maar de talloze keren dat ik je deur platliep (of eerst nog de deur

[1]Ter mijner verdediging: Wachtlijntheorie overlapte met een keuzevak waar we met amper 5 studenten zaten.

van het labo), stoorde ik zelden. Waar je ook mee bezig was, het kon altijd wel wachten. Bij een tegenslag of een non-resultaat werden meermaals de geruststellende, gevleugelde woorden "*Ook dat is onderzoek.*" gesproken. Ze zullen me altijd bijblijven en ik heb ze ondertussen al meermaals zelf gereciteerd. Joris, ik kon me geen betere promotor en mentor wensen.

Aansluitend bedank ik graag de TELIN collega's die het leven hier op de vakgroep kleur hebben gegeven. Bij uitstek de mensen waarmee ik het *io io ee*-bureau door de jaren heen gedeeld heb: Bart, Julie, Kurt, Lennert, Huynh, Adriaan, Hannah. Ik kom graag nog eens meehelpen als er personeel te kort is in *Santa's workshop* of de tuinkers water moet krijgen in iemands afwezigheid.

Verder bedank ik nog graag Patrick en Sylvia, temmers van het administratieve monster en altijd goed voor een praatje en een woord van steun.

En last but not least onze systeembeheerders Philippe en Davy. In mijn zes jaar op de vakgroep kan ik de technische pannes op één houtzagershand tellen. Ik ken genoeg voorbeelden waar dat anders is. Ik heb alvast medelijden met alle systeembeheerders op mijn toekomstig carrièrepad, want ze zullen afgemeten worden aan de standaard die jullie hebben gezet. Ook de vele middagen waarbij we het 13u journaal van commentaar en kwinkslagen voorzagen, heb ik altijd leuk gevonden.

Mijn ouders en mijn familie mogen in dit dankwoord uiteraard niet ontbreken. Mama, papa, jullie hebben me altijd alle kansen gegeven en gezorgd dat ik niks tekort kwam. Jullie onvoorwaardelijke steun en hulp in alles wat ik doe, is van onschatbare waarde. Zonder jullie stimulatie om altijd mijn best te doen en niet te snel tevreden te zijn, was ik hier nooit geraakt.

Ook mijn grootouders, die altijd zeer nabij aanwezig waren, hebben me mee gevormd als mens. Germaine, Esther en Paul — mij beter bekend als dichte meme en verre meme en pepe; ver want ze woonden immers 2,5 km verder terwijl dicht slechts 1 deur verder was — het is jammer dat jullie dit moment niet meer kunnen meemaken. Dichte pepe, Jerome, godfather van het Vanlerberghe geslacht, ik heb veel van je geleerd en ik hoop dat ik nog lang bij je kan komen binnenvallen voor een praatje. Om op jouw leeftijd nog zo kwik van geest en best goed te been te zijn, teken ik direct. Dat ik daarvoor mijn haar moet verliezen, neem ik er zonder meer bij. Pepe, nu moet ik echt niet meer naar *school* en die zomervakantie waar je vanuit ging dat ik ze altijd had, heb ik nu zeker niet meer.

Mijn broers Jeroen en Ruben, we verschillen wel eens van mening en in combinatie met de koppigheid die ons als Vanlerberghes ty-

peert, is dat niet altijd een succesverhaal. Ik weet echter dat jullie er altijd voor me zullen zijn, en ik hoop dat jullie weten dat je hetzelfde kunt verwachten van mij. Enig kind zijn was voor niemand van ons iets geweest, denk ik. Het feit dat ik nu goed kan discussiëren en argumenteren heb ik ongetwijfeld ook aan jullie te danken.

Dan heb ik nog mijn vrienden waarvan ik 95% tegen het lijf ben gelopen in chiroverband. De Chiro, ik heb er veel van gekregen en geleerd. Dat probeer ik intussen al jaren ook allemaal terug te geven; ik blijf er echter ook veel van terugkrijgen. When does it ever stop? Vroeg of laat kruipt het onder je vel en wordt het een stuk van jezelf. In het bijzonder aan het organiseren van Aspitrant/Aspibivak, het bouwen van touwenparcours en van spectaculaire constructies op startdagen en Krinkels heb ik enkel goede herinneringen. Ook de vele gezellige voorbereidingsweekends in het bos met de Tochtenbivakploeg zorgden altijd voor de nodige ontspanning. Ik hoop van harte dat we blijven projecten en excuses verzinnen om regelmatig bijeen te komen, al hebben we in feite geen excuses nodig.

Tenslotte wil ik nog een aantal vrienden extra bedanken. Joke, het heeft even geduurd voor we elkaar gevonden hadden op verschillende evenementen, maar ik ben heel blij dat ik je uiteindelijk toch leerde kennen. Wij komen alvast een tikkeltje dichter wonen dan kunnen we wat vaker bij elkaar binnenvallen.

Hannes en Simon, ik ben blij dat we na onze engagementen in de chiro elkaar ook hebben gevonden in een gezamelijke passie, voor klimmen, engineering en trash-talk. Een gemeenschappelijk moment prikken is een heuse uitdaging geworden. Het resultaat is dat we al eens sneller een hele tijd zeveren, ik bedoel bijpraten, dan echt toe te komen aan het expliciete doel van de afspraak. Echter die berg beklimmen het komende jaar: we maken er werk van!

Als laatste Jesca, ik denk dat je uit eerste hand kunt getuigen dat het afwerken van een doctoraat niet echt een feestje is. Het samenleven met iemand die dat aan het doen is, evenmin. Ik weet hoe moeilijk ik kan zijn als ik wat onder stress sta. Bedankt dat je het ook altijd in die context hebt kunnen plaatsen en me hebt geholpen door andere dagelijkse beslommeringen uit handen te nemen.

Aan jullie allen, ik had de laatste maanden niet altijd even veel tijd voor jullie, mijn excuses. Maar tijd voor een feestje, nu. Een feestje voor jullie evenzeer als het er eentje voor mij is.

Gent, oktober 2018
Jasper

Elk van de hoofdstukken start met een uitspraak of spreuk die ik al meermaals mocht aanhoren van mijn helden. Bij deze laat ik jullie graag mee genieten van hun wijsheden die me mee gevormd hebben.

Table of Contents

Nederlandstalige samenvatting

Wachten leren we van jongs af aan, maar het blijft even irritant. We kennen wachten vooral van de fysieke wachtlijnen in de supermarkt, aan de bankautomaat en op de autostrade. Er zijn echter ook minder zichtbare wachtlijnen. Ook een bufferende video, een verhakkeld telefoongesprek en een traag ladende website zijn meestal te wijten aan datapakketjes die ergens in een wachtlijn op een server staan. Aangezien we allemaal niet graag wachten is het voor veel bedrijven interessant om dat wachten te vermijden of toch ten minste de wachttijd zoveel mogelijk te beperken. Het is immers over het algemeen niet verstandig om het geduld van de klant op de proef te stellen. Anderzijds kan het wachten ook voor economisch verlies zorgen, als bijvoorbeeld machinewisselstukken ontbreken of je dure werknemers ergens in een file staan.

De wachtlijntheorie heeft net als doel om deze wachtlijnen wiskundig te modelleren. De resulterende modellen worden vervolgens uitgebreid bestudeerd. Het resultaat van deze studie verschaft meer inzicht om met kennis van zaken de meest efficiënte ingrepen te doen om het wachten in de bovenstaande voorbeelden te beperken. Die efficiëntie wordt niet enkel bepaald door de beperking van de wachttijd, maar ook de te investeren kost en eventuele andere beperkingen. De wachtlijntheorie situeert zich op die manier binnen de doorsnede van de toegepaste wiskunde en het operationeel onderzoek.

In dit proefschrift focussen we specifiek op wachtlijnsystemen waarbij verschillende types of klassen entiteiten gebruik willen maken van dezelfde service. De entiteiten worden opgedeeld onder deze types of klassen omdat ze een verschillende prestatie verlangen van het systeem. Tegenwoordig wordt op dezelfde telefoonlijn of kabel een heel diverse set aan services aangeboden: vaste telefonie, internet, digitale tv en business services. Dat je e-mails iets later verstuurd worden of de synchronisatie van je bestanden naar de cloud iets langer duurt kan je gemakkelijker aanvaarden (als je het zelfs al merkt), dan dat je televisiebeeld hapert of je telefooncommunicatie spaak loopt door vertragingen en onderbrekingen. Veel bedrijven betalen extra voor een business service, daartegenover staat een gegarandeerde minimum-kwaliteit van het netwerk. Het is voor de provider dus belangrijk om op een zo kosten-efficiënt mogelijke manier de verschillende kwaliteiten voor deze aangeboden services op hetzelfde netwerk te kunnen realiseren.

Om dit te bewerkstelligen, zal het netwerk de pakketjes van de verschillende services anders gaan behandelen en dus meer of minder voorrang geven. Deze gedifferentieerde service wordt verkregen door het implementeren van een scheduler in de knopen van en toegangen tot het netwerk. Deze scheduler bepaalt in welke volgorde de verschillende pakketjes behandeld worden. De eenvoudigste scheduler in dat verband werkt met strikte prioriteiten. Alle types pakketten worden geordend in een hiërarchie; het type dat het hoogst staat in de hiërarchie heeft de hoogste prioriteit. De scheduler behandelt de aanwezige pakketten dan steeds volgens de plaats van hun type in de hiërarchie.

Het grote voordeel van een prioriteits-scheduler is zijn eenvoud. Het nadeel is echter dat het mechanisme weinig flexibel is: ofwel heeft type A strikt voorrang op type B ofwel omgekeerd, eigenlijk willen we dikwijls iets tussenin. Die strikte vorm van prioriteit kan er ook voor zorgen dat de types pakketten met laagste prioriteit helemaal geen service meer krijgen, wanneer er veel verkeer is met hogere prioriteit. Ook dat is zelden gewenst en zorgt voor een slechte service.

Een flexibeler scheduling mechanisme is het zogenaamde Generalized Processor Sharing (GPS). Bij GPS krijgt elk type een gewicht toegekend, de beschikbare capaciteit van de server / het netwerk wordt dan proportioneel tot dat gewicht verdeeld over de aanwezige types pakketten. Dit mechanisme geeft de netwerkbeheerder een flexibelere manier om het verkeer van de verschillende services op een verschillende manier te behandelen in het netwerk om zo een gepaste kwaliteit voor de desbetreffende service aan te bieden.

In dit proefschrift bestuderen we specifiek een discrete-tijd, probabilistische implementatie van deze GPS scheduler. In de implementatie voor twee types pakketten krijgt type 1, in plaats van een gewicht, een probabiliteit β toegewezen. Als er in de wachtlijn van elk type minstens één pakket aanwezig is, dan wordt met kans β het oudste pakket van type 1 bediend en met kans $1-\beta$ het oudste pakket van type 2. Als er slechts één type pakketten aanwezig is, wordt uiteraard dat type bediend. Merk op dat door een passende keuze van de gewichten of probabiliteiten in een GPS-scheduler men een zuivere prioriteits-scheduler bekomt.

Over prioriteitsmodellen is al veel onderzoek gebeurd en gepubliceerd. De studie van GPS-modellen staat er echter om bekend om vele malen complexer te zijn dan die van prioriteitsmodellen. Hier-over zijn significant minder analytische resultaten te vinden in de wetenschappelijke literatuur. Dit proefschrift vult deze resultaten aan met nieuwe inzichten.

In het eerste hoofdstuk presenteren we een algemeen beeld van de wachtlijntheorie en hoe de bijdrage van dit proefschrift daarin past. We introduceren er uitgebreid de discrete-tijd implementatie van GPS, het onderwerp van dit proefschrift, en de verhouding met de geïdealiseerde continue-tijd versie. Als laatste definiëren we er de belangrijkste begrippen en symbolen die gebruikt worden in de rest van de tekst.

De rest van het proefschrift is opgedeeld in twee delen. Deel I behandelt de optimalisatie en de haalbare prestaties van GPS-wachtlijnen. In hoofdstuk 2 bekijken we eerst de mogelijke prestaties van een systeem met twee klassen. We bewijzen er twee cruciale eigenschappen die we later in dat hoofdstuk gebruiken om enkele belangrijke stellingen over de optimalisatie van het systeem te bewijzen. We bestuderen de optimalisatie van algemene kostfuncties die gedefinieerd zijn in termen van van de gemiddelde resterende werkhoeveelheid van de twee types. Voor kostfuncties die een gewogen som van functies g_j $(j = 1, 2)$ van de gemiddelde resterende werkhoeveelheden van de twee klassen zijn, bewijzen we dat een GPS-scheduler enkel optimaal is als de g_j stijgende convexe functies zijn en niet als ze lineair of concaaf stijgend zijn. In die gevallen is strikte prioriteit optimaal. We eindigen dat hoofdstuk met een uitbreiding van de belangrijkste stellingen naar een meer algemene setting, los van GPS-scheduling. In hoofdstuk 3 bekijken we de uitbreiding van het systeem met twee klassen naar drie klassen. We tonen aan dat een hiërarchische versie met drie klassen verschillende voordelen heeft en hetzelfde prestatiebereik als de niet-hiërarchische versie. We sluiten af met een algoritme om efficiënt de configuratie van de GPS-scheduler te bepalen voor een gewenste prestatie en een bespreking over de uitbreiding naar meer dan drie klassen.

In deel II behandelen we de analyse van GPS-wachtlijnen. In hoofdstuk 4, starten we met de samenvatting van een benaderingsalgoritme uit de literatuur voor het systeem met twee klassen. Dit algoritme levert een methode om iteratief hogere coëfficiënten in de oneindige machtreeks van de gemiddelde resterende werkhoeveelheid te berekenen. In de praktijk is het echter computationeel onmogelijk om veel coëfficiënten te berekenen. Dat is de grootste beperking van die methode. We bekijken hoe we toch het meeste uit deze benadering kunnen halen en bekijken de prestatie ervan in een optimalisatie-setting. In hoofdstuk 5 breiden we dit benaderingsalgoritme uit naar de hiërarchische GPS-versie voor drie klassen en bekijken ook specifiek het geval waarbij het systeem één hoge prioriteitsklasse heeft en de overige capaciteit via GPS verdeeld wordt over de overige twee klassen. Ten slotte, bekijken we in hoofdstuk 6 een alternatieve methode om diezelfde machtreeks te bekomen, zodat we wel veel coëfficiënten kunnen berekenen. Deze methode is hier uitgewerkt voor twee klassen en een eenvoudiger aankomstproces dan in de rest van dit proefschrift, maar het zet belangrijke stappen om de uitdagingen uit hoofdstukken 4 en 5 te overwinnen. We eindigen het proefschrift met een bespreking van de belangrijkste resultaten in hoofdstuk 7.

Summary in English

Even though we learn to wait while we are still in diapers, it stays equally irritating. Waiting is most known from physical queues in the supermarket, the ATM machine or the highway. There are, however, also less visible queues. A buffering video, a disrupted phone call and a slowly loading website are usually also caused by data packets queueing up somewhere in the network. As nobody likes to wait, it is interesting for many companies to eliminate this waiting or at least to try and minimize the delay as much as possible. After all, in general, it is not wise to challenge the patience of the customer. On the other hand, queueing can also mean economical loss if, for instance, spare parts for machines are missing or expensive employees are stuck in a traffic jam.

Queueing theory has the goal to mathematically model these queues. Subsequently, these models are extensively studied. The result of this study provides insight to make efficient adaptations in an informed way to limit the queueing in the previously mentioned examples. This efficiency does not only refer to the limitation of the delay but also to the investment cost and possible other restrictions. That way queueing theory is on the intersection of applied mathematics and operations research.

In this dissertation, we specifically focus on queueing systems whereby several types or classes of entities require service from the same server. The entities are categorized in these types or classes because they desire a different quality of service from the system. Nowadays, the twisted pair or coax cable to our home offers a wide range of services: telephone, internet, digital television and business services. We more easily accept that our e-mail is sent a little later or the synchronization of our files is a little slower (if we even notice it at all), than a television image full of artefacts or a disrupted phone converstaion. Many companies pay extra for a business service that guarantees a minimum quality of the network. In that regard it is important for the provider to have a cost-efficient way to implement service differentiation for this range of offered services on the same network.

To effectuate this, the network will treat the packets of different services differently and thus give them more or less priority. This service differentiation is achieved by implementing a scheduler at the nodes and access points of the network. This scheduler determines the order in which the packets are handled. The simplest scheduler in that regard is a strict priority scheduler. In that case, all classes are ordered in a hierarchy; the class that is on the highest level in the hierarchy has the highest priority. The scheduler then treats the backlogged packets in the order of their class in the hierarchy.

The greatest advantage of a priority scheduler is its simplicity. The disadvantage, however, is that the mechanism has little flexibility: either class A has strict priority over class B or vice versa; while often we desire something in between. This strict form of priority can also cause starvation of the lower priority classes when an abundance of higher priority traffic is present. This behavior is also seldomly desired, and is regarded as bad service.

A more flexible scheduling mechanism is so-called Generalized Processor Sharing (GPS). In GPS each class is assigned a weight and the available capacity of the server / network is proportionally divided amongst the backlogged classes. This mechanism equips the provider with a more flexible way of treating the packets of the various classes differently in the network and thus offering a different quality of service for the various services.

In this dissertation, we study a discrete-time, probabilistic implementation of this GPS scheduler. In the implementation for two classes of packets, class 1 is assigned a probability β instead of a weight. If packets of both classes are backlogged, the oldest packet of class 1 is served with probability β whereas with probability $1-\beta$ the oldest packet of class 2 is served. If only packets of one class are backlogged, that class is served. Note that by an appropriate choice of the weights or probabilities the GPS scheduler becomes a strict priority scheduler.

There has already been much research on priority queueing models. The study of GPS models, however, is known to be notoriously more complex. On these models there are significantly less analytical results available in the scientific literature. This dissertation complements these results with new insights.

In the first chapter, we present a general overview of the queueing theory field and where this dissertation fits in. We introduce in full the discrete-time implementation of GPS that is the subject of this dissertation and the relation to the idealized continuous-time version. Lastly, we define the most important concepts and symbols that are used in the rest of the text.

The remainder of the dissertation is divided in two parts. Part I treats the optimization and achievable performance of GPS queues. In chapter 2 we first study the achievable performance of a system with two classes. We establish two important properties of the system that we will use later in the chapter to prove some theorems about the optimization of the system. We study the optimization of general cost functions that are a function of the mean unfinished work of the two classes. For cost functions that are a weighted sum of functions g_j $(j = 1, 2)$ of the mean unfinished work in both classes, we prove that the GPS scheduler is only optimal if the g_j functions are increasing convex functions and not when the g_j functions are linearly or concave increasing. In those last cases, strict priority is optimal. We end the chapter with the extension of the applicability of the most important theorems to a more general setting, apart from GPS scheduling. In chapter 3, we look at the extension of the two class system to three classes. We show that a hierarchical version has several advantages and has the same achievable performance range as the non-hierarchical version. We end with the presentation of an algorithm to efficiently determine the configuration of the GPS-scheduler to achieve a desired performance and discuss the extension of the system to more than three classes.

In part II, we treat the analysis of GPS queues. In chapter 4, we start with a summary of an approximation algorithm for the two-class system, that we retrieved from the literature. This algorithm provides a method to iteratively calculate higher-order coefficients of the infinite power series of the mean unfinished work. In pratice, however, it is computationally impossible to calculate a large amount of coefficients. That is the major limitation of the method. We look at how we can still get the most out of this method and look at the performance thereof in an optimization setting. In chapter 5, we extend the approximation algorithm to the hierarchical GPS version with three classes and also specifically study the special case with one high-priority class and whereby the remaining capacity is divided using GPS amongst the other two classes. Lastly, in chapter 6, we present an alternative method to calculate the same power series, that makes it possible to calculate many more coefficients. That method has been developed for two classes and a simpler arrival process than in the rest of this dissertation, but it contains important steps to overcome the challenges from chapters 4 and 5. In chapter 7, we end the dissertation with a discussion of the most important results.

"Je moet niet denken dat de gebraden kippen in je mond gaan vliegen."
— Willy Vanlerberghe

1 Introduction

1.1 Queues and Queueing theory

Queues are everywhere. Some are very visible and annoying — especially when you are in (the wrong) one. The classical example is lines at the supermarket's cash registers. Others are less visible — but equally annoying — such as the buffering of a Youtube video or a disrupted phone conversation.

Queueing theory is a mathematical discipline studying systems in which certain *entities* require a certain *service* that takes some time to execute. This real-life situation is formalized and simplified in such a way to end up with a mathematical model of the queueing system. Firstly, the arrival process describes the way in which the entities requiring service, arrive at the system, e.g. the arrival rate. Secondly, we have some waiting and scheduling rules. The waiting rules describe for instance how much waiting room is available (if any) and what happens when entities arrive when no room is available. The scheduling rules describe in what order entities are handed over to the service area. The system can choose to categorize the entities based on the kind of service they require, and enqueue them each in a separate queue. Next, we specify the service area that can typically have one or multiple *servers* which are delivering the service to the entities. Lastly, we have the service process that describes the speed at which the entities are served. Roughly speaking, these inputs de-

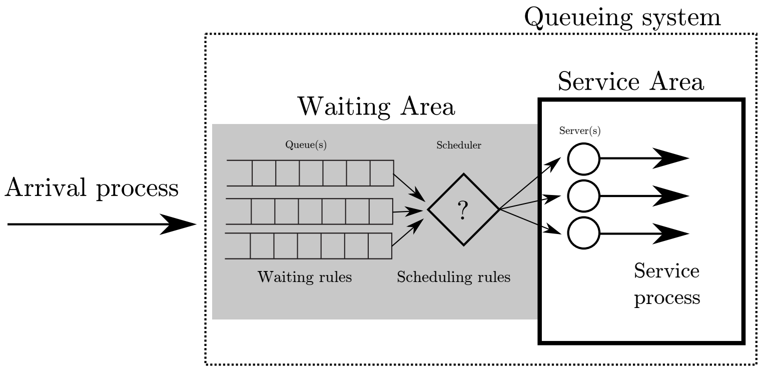


Figure 1.1: Mathematical model of a queueing system

scribe the mathematical model. We have summarized these elements in Figure 1.1 and we elaborate on each of them in the subsequent subsections. [1]

1.1.1 Arrival process

The arrival process describes how the identities arrive to the queueing system. Depending on the applications, these arriving entities are often called *customers*, *jobs* or *packets*. This is also the case in the remainder of this dissertation. The description of these arrivals can for instance be a specification of the *interarrival time* between customers. The interarrival time can be either (1) deterministic, i.e., following a predetermined pattern for instance always the same length, or (2) stochastic, i.e., the length of the interarrival time is dictated by a stochastic process. In many cases the interarrival times of a stochastic process are specified to be independent and identically distributed (i.i.d.). This basically means that the interarrival time between arrival i and arrival $i+1$ is independent of all previous interarrival times and the probability distribution of its length is identical to that of all other interarrival times. This independence makes things much simpler, as for the generation of arrivals to the system no information about the previous arrivals needs to be kept. More complicated (not

[1]In this very general description, we have totally ignored the subdiscipline in queueing theory that studies networks of the latter described models. These queueing networks are no subject of this dissertation and thus for conciseness we omit further elaboration.

i.i.d.) stochastic processes are used in queueing theory as well, but are not of interest for this dissertation.

Further details of the arrival process can be the arrival of the customers in batches [38, 39]. The size of these batches can again be either deterministic or stochastic. Another possible specification of the arrival process is the distinction between several *types* or *classes* of customers. The distinction between these customers is made for use inside the queueing system. For instance, they require a different kind of server, have a different service requirement (e.g. longer service) or are handled differently by the scheduler.

1.1.2 Waiting area

The description of the waiting area of the queueing system specifies what happens with the customers after their arrival. In the simplest case there is no real waiting area, i.e. a customer that arrives to the system does not get service and is *lost*. This is called a *pure loss system*.

> In fact, one of the first ever studied queueing systems was a pure loss system. At the start of the twentieth century, the Danish mathematician Agner Krarup Erlang was working at the Copenhagen Telephone Company. In those days, telephone calls were switched manually by operators at an exchange using jack plugs. While working at the telephone company, he tried to solve the classic problem of determining how many lines were needed between two exchanges to provide an acceptable service. After all when all lines are in use, no more calls can be relayed between the two exchanges, thus blocking the call. This problem sparked the start of queueing theory, when Erlang presented among others the famous Erlang-B formula [56]. This formula expresses the probability of blocking an incoming call because all lines are in use. In this Erlang B model there are n servers, which model the number of lines between the two exchanges. The arrival process models the incoming calls for which one of these lines/servers are needed. There is no waiting area or queue. When a call arrives it either gets one of the available lines, or when no lines are available the call is blocked and the revenue of the customer is lost.

In other cases, customers are defined to arrive in a queue, whereby the queue has a maximum number of customers it can hold. Customers arriving when the queue is full are lost [51]. From a theoretical perspective, queues which can hold an infinite number of customers are also interesting as they can be easier to study. This is also the case in this dissertation. Lastly, the waiting area can also have multiple queues which are used to backlog customers of a certain class or type.

1.1.3 Scheduler

The scheduling rules are imposed by a — not always explicit — scheduler. The simplest and most well-known scheduling rule is First Come First Served (FCFS). One can either have global FCFS, whereby the next packet chosen to be served is the *oldest* packet in the waiting area; and in case there is only one queue also the oldest packet in that queue [104]. On the other hand, when the waiting area has several queues for several types of packets, it is also possible for the scheduler to first choose a queue and to then take a packet from that queue on a FCFS basis. In that case the packet chosen is not necessarily the oldest packet in the waiting area.

In case the model has several classes of packets that are backlogged in their own queue, the queueing system can employ a priority scheduler. In this model a priority hierarchy between the different types of packets is specified. The scheduler, subsequently, schedules the packets according to their class priority order [141,144]. This priority scheduler is very strict, a possible variation to make the scheduling more flexible, is a model whereby packets from a certain class can *jump* to a higher class queue under certain conditions [102]. In this dissertation, the main focus is the study of a certain type of scheduler, that is related to the aforementioned priority scheduler that also aims at providing more flexibility. We postpone the full specification of this scheduler to the next section after this general introduction on queueing theory.

Some other types of models have been studied whereby customers in the queue are served in batches, which is for instance applicable for guided tours in a museum that only start when enough people are present [37]. In another type of model with several types of customers the scheduler waits for one customer of each type to be in the queue before scheduling each of them simultaneously for service. This kind of model is for instance used when studying factories, whereby products can only be assembled if all parts are available [46].

1.1.4 Servers and Service process

As a last element of Figure 1.1, we discuss the servers and the service process. A queueing system can have one or more servers. Lastly, we have the service process, this is a specification of how long it takes the server to serve a certain customer. This *service time* can again be either (1) deterministic or (2) stochastic. A different service process can be specified for different classes of jobs (some classes of jobs require more time than others) or for different servers (some servers are *better* or *faster* than others).

Once again many variations are possible and have been studied. One can introduce server vacations or working vacations whereby one or more of the servers are disabled or work at a lower rate [64, 84]. There are models whereby some of the servers can only handle a specific class of customers and the rest of the servers can do all types of work [68, 122]. These models are used to study applications with some task-specific machines/people and other general purpose machines/people. Another variation that has been studied for applications with real-world computer servers, is a model whereby unemployed servers speed up the service of other servers.

1.1.5 Continuous time vs Discrete time

We study the evolution of the queueing system in time. For the mathematical analysis of this queueing system, we can look at time as either continuous or discrete. *Continuous time* is time like we know it in real life; at any given instant a customer can arrive or leave the queueing system. In this case, the time variable is continuous and can thus take any value on the positive real axis. Likewise, the interarrival time between two customers or the service time of a customer are continuous random variables and thus real-valued.

In case the queueing system is defined as a *discrete time* system, time is considered to be slotted. This means time is divided in equal length parts, called slots. Subsequently, the system is studied on these slot boundaries. It is only on these slot boundaries that changes in the system happen, for instance the arrival/departure of customers or the service start for a certain customer. This mathematical representation of time stems from digital (computer) systems where all events are synchronized. These systems have an internal *clock* that generates the slot boundaries; only at these boundaries the state of the machine changes. The clock speed, e.g. 3GHz, of a computer processor indicates that it has 3 billion slots in one second. As a result of slotted time, we number the slots and time thus becomes a discrete value, meaning it can only take on certain values, namely the set of integer numbers. Furthermore, we usually do not describe the interarrival time between customers but the number of customers arriving in a slot. The service time of a customer is now a certain number of slots. The arrival process and service process are thus described using discrete random variables.

Continuous and discrete-time systems are studied using different mathematical techniques, although there are similarities. From an application point of view, modeling an application and studying it in discrete time vs continuous time should of course lead to similar results.

1.1.6 Performance measures

In the previous subsections, we have addressed the inputs of the model. In this subsection, we discuss the possible outputs of the study of the model. These outputs are in general some *performance measures* about the behavior of the system. On the one hand, there are system specific performance measures, that say something about the state of the system. For instance, we have the number of customers in (each of) the queue(s), i.e., the *queue content*, or the complete system. As each customer requires a certain amount of service time (possibly different from customer to customer cf.: stochastic service times), likeso we can also quantify the *unfinished work* in the queue(s) or the system. This unfinished work expresses the amount of service time necessary to empty the system, given that no more customers arrive. On the other hand, there are customer-specific performance measures, like the queueing time or *waiting time* of the customer. When you add the service time of a customer to its waiting time, you get its system time or *delay time*.

1.1.7 System stability and Steady state

The performance measures discussed in the previous subsection are random variables as they depend on the inputs (arrivals and service time) that are also random variables. For each point in time or each packet (depending on the performance measure) the random variable is different. For instance in a single-server discrete-time system that started with 10 customers requiring a single slot of service, the probability of having an empty queue is 0 for the first 10 slots but then increases in the later slots[2]. This is of course non-practical as different performance measures for every epoch / customer make it hard to make statements about the performance of the system in general.

However, when the rate at which work enters the system (*load*) is smaller than the rate/capacity at which work can leave the system, the system will typically evolve to an equilibrium. This is not the kind of equilibrium whereby the amount of work in the system is constant, but rather the probability distribution of the amount of work in the system is constant in time. This equilibrium is called the *steady state* of the system and the system is said to be *stable*. The

[2] Provided that the system is not in *overload*, i.e. the stability condition is met (see further in this paragraph).

condition whereby the offered load is smaller than the capacity of the system is called the *stability condition.*

1.1.8 Model specification vs analysis

In the previous subsections, we addressed the problem setting of queueing theory in general[3]. Specifically, we introduced some key parts, variables and conditions for the specification of the queueing model. This queueing model can be as *complicated* as the author of the model wants and is only limited by her/his imagination.

The specification of the model, however, is only a starting point. Deriving *interesting* results, i.e. the analysis, is a different matter. The analysis of the model is usually the difficult part and where the researcher spends most of his time on. For this analysis several methods have been developed. Interestingly, seemingly small changes to the model, can render the *old* method unusable and the problem (temporarily?) insolvable.

In the remainder of this chapter, we introduce the model studied in this dissertation. Or at least we introduce the general setting, as small variations exist between the models in the different chapters. Furthermore, we explain the goal of the analysis and describe the structure of the rest of the text.

1.2 Generalized Processor Sharing

This dissertation focuses on the analysis of the performance and optimization of a certain class of scheduling disciplines. In this section, we take a look at the motivation for these kinds of scheduling disciplines and their history. In the next section, we rigorously define the discipline that is the subject of this dissertation.

The most widely known scheduling discipline is global FCFS. All customers are treated equally and are served in the order of their arrival. The policy is well known for its fairness. However, there are applications and situations, where we do want to tamper with the equal treatment of customers. For instance at an emergency room, it feels natural to discard FCFS to postpone the treatment of a broken ankle for a newly arriving car crash victim with a polytrauma. Another example comes from communication network providers. These commercial companies often offer premium packages to business users, to provide them with higher quality service. In these cases their net-

[3] albeit not aimed at providing a full overview or definition of the field.

works have to be able to deliver this premium service and differentiate the handling of packets at the nodes in the network.

A straightforward solution is to classify all incoming customers (or packets) in a number of service classes each having their own queue. Additionally a priority ordering is made amongst the classes. The server subsequently serves the oldest customer (i.e.: with earliest arrival time) of the queue holding the customers with the highest priority present in the system. This policy, known as (strict) priority queueing, solves the issues of the previous examples. Priority queueing, however, suffers of possible starvation of some of the classes [75,103,141]. For instance, assume the nodes in a communication network give priority to the traffic from business users. If these business users generate enough traffic to occupy the entire outgoing bandwidth of the node, the traffic from regular users would never be sent. The service of these users is completely degraded by the *misbehavior* of higher-class users. This starvation problem can also occur to a smaller degree with global FCFS in the nodes of the communication network. If a certain user or users inject large amounts of traffic some other users can temporarily be starved of service by the network. This starvation problem is unacceptable for service providers (because it is unacceptable for its users). A solution to the problem is to have some sort of rate admission control [67]. This shapes the ingress of traffic of a user to the network at its access point to not *destabilize* the service of the network. However, this flow control is not always possible and could possibly be circumvented by mischievous users hacking the software of their access point.

The drastic difference in quality of service offered to the classes in the queueing system causes the problem of starvation. Additionally, this kind of drastic difference might a priori not be what is desired. An alternative is Generalized Processor Sharing or GPS. GPS is an idealized scheduling discipline specified by Parekh and Gallager in their seminal papers [109, 110]. In GPS the customers arrive and are classified in classes or flows, each with their own queue. Every class is assigned a certain weight, whereby the weight of class i, i.e. ϕ_i, is a positive real number. Time is continuous and if all queues are backlogged queue i is served at a rate of $\frac{\phi_i}{\sum_j \phi_j} r$, whereby r is the maximum rate at which the processor can process work. As such, the different queues are served at a rate proportional to their weight. In case not all queues are backlogged, the freed up capacity is redistributed amongst the non-empty queues proportionally to their weights.

GPS avoids the problem of classes with high or bursty traffic blocking others and guarantees a minimum share of the bandwidth to each customer class. Simultaneously, it provides a means to differentiate service amongst customer classes in a flexible and continuous way. However, GPS is mainly a theoretical model, as it is an idealization. For most real-world systems it is impossible for a single server to serve more than one (class of) customer(s) simultaneously. Generally the work for different customers needs to be interleaved in time. As such, one customer would be served up until its quotum is used up after which service is switched to a next customer. By interleaving the work with quota approaching zero, in the limit GPS is achieved. This is of course not useful in practice and one wonders if real-world implementations aiming at approximating GPS yield different results performance-wise.

In the next section, we specify a discrete-time implementation of GPS that is no longer an idealization. This implementation is a candidate for real-world applications. In the subsequent chapters, we study the performance and optimization of this discrete-time GPS in detail.

What is exactly *Generalized* about this Processor Sharing?
Generalized Processor Sharing is a generalization of (Uniform) Processor Sharing [89]. Processor Sharing in turn was developed as a model for time-sharing on computer processors. The processor of a computer — or nowadays better a single core of a processor — has several processes waiting for service. For a fixed time quotum a process is given service after which a context switch is done, and the next process receives service. By letting the quotum size approach zero, in the limit, Processor Sharing is achieved. As such each process receives an equal time share of the processor. If in the definition of GPS one chooses all ϕ_i equal and defines every process to be a different class, it is effectively Processor Sharing. In that way GPS is a generalization of Processor Sharing providing a greater flexibility to fine-tune the service differentiation.

1.3 Discrete-time implementation of Generalized processor sharing

After the previous introductory sections, we now present the specific scheduling studied in this dissertation. As mentioned before, it is a discrete-time implementation of Generalized Processor Sharing. We start by looking at a system that has only two[4] different customer classes. In Chapter 3, we study the extension to three classes and more.

[4]Clearly if the system has only one customer class, no service differentiation and thus scheduling between classes is needed.

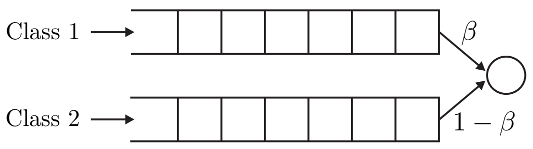


Figure 1.2: Figure of model for two customer classes

The arriving customers are classified in two separate classes, that are each backlogged in their own queue. These queues are assumed to have infinite storage capacity. The specification of the arriving process defines the arrival rate of each of the classes and the correlation between both. In this dissertation[5] we assume a general two-dimensional arrival process that is independent and identically distributed from slot to slot (cf. subsection 1.1.1). Furthermore, we assume every customer requires a single slot of service.

When both queues are backlogged, the server serves a customer of class 1 with probability (w.p.) β and a customer of class 2 with probability $1-\beta$ ($\beta \in [0,1]$). When only one of both queues is backlogged and the other is empty, the non-empty queue is served. As a result, the scheduling is so-called work conserving, which means whenever work is present in the system the server is never idle. This is summarized in Figure 1.2.

Special cases When $\beta = 1$ this GPS scheduling equals a strict priority system with high priority for class 1 and low priority for class 2. When $\beta = 0$ it is the other way around: high priority for class 2 and low priority for class 1.

Symmetry This scheduler is symmetric in β. The system in Figure 1.2 provides an identical performance to class 1 as Figure 1.3 and dito for class 2. With a sufficiently symmetric arrival process, which is the case for arrivals that are i.i.d. from slot to slot, results for parameter $\beta = \beta^*$ can be used to obtain results for $\beta = 1-\beta^*$ by enqueueing class-1 (class-2) customers in queue 2 (queue 1).

The analysis of GPS queueing systems, however, is much harder than that of priority queues. For priority queues a wide range of systems have been analyzed in the literature [51, 102, 141, 147]. For GPS queues the results are mostly limited to upper or lower bounds

[5]Except in Chapter 6 where a more restrictive arrival process is used.

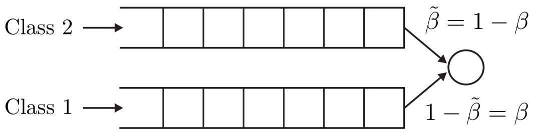


Figure 1.3: Figure of symmetric model

on the performance, or the presented methods require a significant numerical effort [41,69,110,154]. In this dissertation we complement the already available results. In Part I, we focus on the optimization of GPS queueing systems. In Part II, we study the analysis of the system and present some approximations for the performance measures.

1.4 Definitions

We study the system in discrete time, which means time is divided in slots and studied at slot boundaries. Slots are numbered using the natural numbers, starting from 1. We define the random variable $a_{j,k}$ as the number of arrivals of class j in slot k and $\boldsymbol{a}_k = (a_{1,k}, a_{2,k}, \dots)$, whereby the size of the vector depends on the number of classes in the system[6]. The number of arrivals of class j in a random slot is written as a_j. Furthermore, we denote the arrival rate of class j by λ_j, i.e. the mean number of class j packets that arrive per slot, and $\lambda_T = \sum_j \lambda_j$ as the total arrival rate. For convenience, we also define $\alpha_j = \frac{\lambda_j}{\lambda_T}$ as the fraction of type-j arrivals in the overall traffic mix.

The random variable $w_{j,k}$ denotes the unfinished work of class j present in the system at the beginning of slot k, i.e.: the number of slots needed by the server to finish all the work of class j when class j is given strict priority. Analogously as for the arrivals, we note $\boldsymbol{w}_k = (w_{1,k}, w_{2,k}, \dots)$. We denote the probability of event A occurring by $\Pr[A]$, equivalently $\Pr[w_{j,k} = l]$ denotes the probability that the unfinished work of type j at the beginning of slot k equals l slots. As introduced in Section 1.1.7, if the stability condition is fulfilled the system will evolve to a steady state. In this regard we, for instance, note w_j as the unfinished work of class j at the beginning of a random slot in steady-state regime. Analogously, we use u_j to denote the number of customers of class j in the queue at the

[6]For these definitions we consider a general number of classes, as we will extend the GPS-model from Section 1.3 to more classes in Chapter 3.

beginning of a random slot[7] and d_j for the delay of a random class-j customer, both in steady state. Furthermore, we write the mean of a random variable with a bar on top or by enclosing it inside the square brackets of $\mathrm{E}[\cdot]$. For instance the mean unfinished work of class j at the beginning of a slot is $\bar{w}_j = \mathrm{E}[w_j] = \sum_{l=0}^{\infty} l \Pr[w_j = l]$. Lastly, w_T denotes the total unfinished work at the start of a random slot in steady state.

In this dissertation, we frequently use probability generating functions (pgf) of the stochastic variables defined above. We denote them with capital letters and use the formal variable z. For instance, $A_1(z) = \mathrm{E}[z^{a_1}]$ is the pgf of the number of arrivals of class 1 in a slot and $A(z_1, z_2) = \mathrm{E}[z_1^{a_1} z_2^{a_2}]$ the joint pgf of the number of arrivals of class 1 and 2 in a slot. From the pgf we can easily calculate the mean of the random variable, for instance, $\lambda_1 = A_1'(1)$.

With these definitions of symbols, we can formalize the two-class discrete-time GPS system from the previous section by writing down the system equations.

- if $\boldsymbol{w}_k = \boldsymbol{0}$

$$\boldsymbol{w}_{k+1} = \boldsymbol{a}_k \tag{1.1}$$

- if $w_{i,k} > 0$; $w_{3-i,k} = 0$ with $i = \{1, 2\}$

$$\begin{aligned} w_{i,k+1} &= w_{i,k} - 1 + a_{i,k} \\ w_{3-i,k+1} &= a_{3-i,k} \end{aligned}$$

- if $w_{j,k} > 0$ for all $j = \{1, 2\}$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (1,0) + \boldsymbol{a}_k \quad \text{w.p.} \quad \beta \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,1) + \boldsymbol{a}_k \quad \text{w.p.} \quad 1-\beta \end{aligned}$$

1.5 Simulations used in this dissertation

A simulation of a queueing system is in fact a *solution method*. In a simulation the queueing system is replicated in a computer program

[7] In Section 1.3, we made the assumption that every customer requires a single slot of service. In that case the unfinished work of class j indeed equals the queue content of queue j, which makes this definition superfluous. However, some results in this dissertation extend to certain cases with customers requiring a stochastic number of slots of service, as we will discuss at the appropriate time. In those cases the queue content no longer equals the unfinished work, which is why we already define it here.

as if it were real. This means that in each slot arrivals are generated according to the specified arrival process and customers are served as specified by the system. At each slot the relevant performance characteristics are recorded, for instance the unfinished work, and at the end the mean performance characteristics can be calculated from the record.

The problem with simulations is that it is a costly operation, it takes the computer a *significant* time to come up with the answer. And this answer is only valid for the specified input configuration. This means that the simulation needs to be repeated for a different β or a different arrival process or configuration thereof. In this dissertation, the main focus is to look for analytic ways of calculating or approximating the performance characteristics, so it is less computationally intensive to change one of the inputs. This also makes it possible to do sensitivity analysis on these inputs and as such optimize the system. However, to establish the validity of our calculations or approximations, we compare them with simulation results.

Providing accurate simulation results in itself is not a given, the simulation result depends on many factors. The first one is the transient period. We want to know the performance characteristics in steady state, however we cannot start a simulation directly in steady state regime, nor is it simple to determine from which point onward the system is in steady state. For simplicity we thus include the transient period (from the start of the simulation to steady state regime) in the calculation of the performance characteristics which introduces a bias in the result. To lessen the bias of the inclusion of the transient period in the result, we can do a longer simulation, i.e., simulate a larger number of slots. This length of the simulation is a second factor that influences the result.

To test the bias introduced by the transient period we ran simulations over increasingly longer periods. For each simulation length, we ran 10 simulations starting from a different starting point. The starting point was determined by drawing two random variables from a uniform distribution of integers in $[0, 20]$, these were subsequently used as the queue contents at the start of the simulation. In Figure 1.4, we display the mean of $\bar{w}_1$ over these 10 simulations versus the simulation length; the errorbars indicate the standard deviation. Starting from a different initial state (queue content at slot 0), results in a different transient period before reaching steady state. We can see that by increasing the number of simulated slots the variance introduced by these different transient periods decreases. This is because the relative length of the transient period compared to the entire simulation decreases.

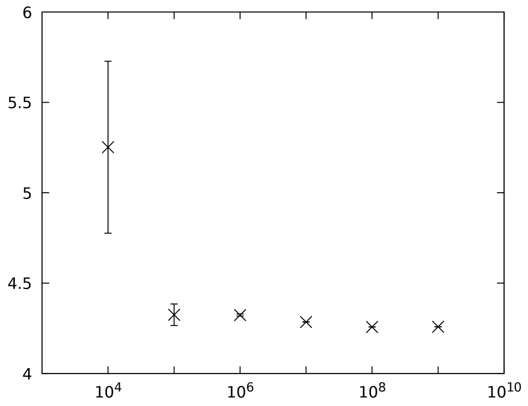


Figure 1.4: Mean of $\bar{w}_1$ with errorbars displaying the standard deviation over 10 simulations with different starting points versus the number of simulated slots.

A last factor is the *choice* of random sequences. In each slot of the simulation a draw from the probability distribution of the arrival process has to be done to generate the number of arrivals in that slot. In a computer this is done using the Monte-Carlo method which converts draws from a uniform pseudo-random sequence generated by the computer to draws from the required (not necessarily uniform) distribution. These pseudo-random sequences look random but are in fact deterministic. They require a seed as a starting point: if the same seed is used the subsequent draws from the sequence are exactly the same. This means that by choosing a different initializing seed, the pseudo-random sequence and the resulting arrivals are different. As such the choice of the underlying random sequence for the arrivals introduces variance in the simulation result.

The same problem arises for the probability to decide which queue to serve. As such the simulation again uses a pseudo-random sequence of numbers (albeit a different one) to have the simulated server make its decisions. The choice of this decision sequence also introduces variance into the result.

To test the variance introduced by the choice of the pseudo-random sequences, we ran several tests. In a first test we chose 10 different arrival sequences, whilst keeping the decision sequence identical and starting from an empty system at the start of the simulation. We repeated this for several simulation lengths. On the left of Figure 1.5, we display the mean over the 10 simulations for $\bar{w}_1$ and

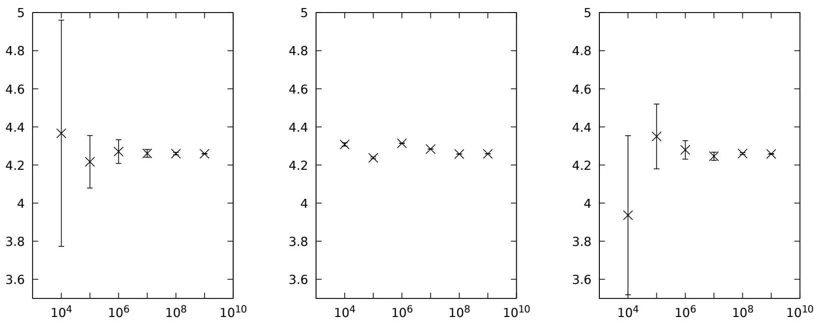


Figure 1.5: Mean of $\bar{w}_1$ with errorbars displaying the standard deviation over 10 simulations versus the number of simulated slots. On the left only the arrival sequence is varied, in the middle only the decision sequence, on the right both arrival and decision sequences are varied.

the errorbars indicate the standard deviation. Analogously we varied the decision sequence for identical arrival sequence and starting state. These results are shown in the middle of Figure 1.5. Lastly, we varied both the decision and arrival sequences simultaneously whilst still using an empty starting state. These results can be found on the right of Figure 1.5.

From Figure 1.5, we conclude that the more slots are simulated the less the choice of the particular sequences matters. For 10^7 slots the standard deviation is only 0.5% of the mean, this further decreases to 0.1% for 10^8 and 0.04% for 10^9 simulated slots. In the remainder of this dissertation, we start all simulations from an empty system and only do one single simulation over at least 10^8 slots[8]. As shown, this simulation length is long enough to provide sufficiently unbiased and variance-free results.

Even though the variance of the simulations is low, it is still present. If we simulate a GPS system with a particular arrival process for 101 β values evenly distributed in the interval $[0, 1]$, we get *non-smooth* graphs. This can be seen in the left graph of Figure 1.6, where we draw $\bar{w}_1(\beta)$ resulting from simulations over 10^8 and 10^9 slots. We show only a subset for $\beta \in [0, 0.5]$ to be able to provide enough detail. This non-smoothness is the aforementioned variance aggravated by the interplay between the small difference in β value and the different arrival and decision trajectory. This interplay is

[8]We either use 10^8 slots or 10^9.

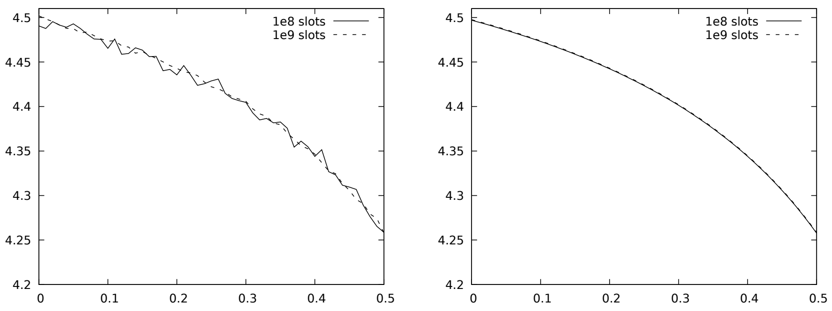


Figure 1.6: Mean unfinished work of type 1 as a function of β. On the right without CRN on the left with CRN.

another source of variance. Together these variances *hide* the true effect of the change of β and can show improper artefacts like local extrema[9].

To solve this problem we use a technique known as *common random numbers* (CRN) or *correlated sampling* [8, 124]. This means that we use identical arrival and decision sequences for all simulated β values. This isolates the influence of this β parameter which is the subject of the search, and strips any variance between simulations from the specific choice of random sequences. Using CRN introduces bias in the result because of this specific choice. However, by choosing a long trajectory for the simulation we have already shown that this bias is sufficiently small.

1.6 A common arrival process

In this dissertation we have frequently used a multinomial distribution, which is a generalization of the binomial distribution to accommodate categories of successes, or, in our case, customer types. In that regard the M-dimensional multinomial distribution is also called a M-dimensional binomial distribution. A M-dimensional multinomial distribution is characterized by the following joint pgf

$$A(z_1, \dots, z_M) = \left(1 + \sum_{j=1}^{M} \frac{\lambda_j}{N}(z_j - 1)\right)^N ,$$

[9]We prove in Section 2.1 that the result should be monotone and thus should not contain local extrema.

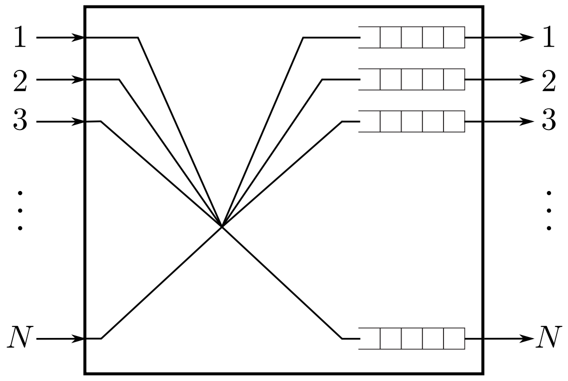


Figure 1.7: $N \times N$ output queueing switch

wherein N is a parameter representing the maximum total number of arrivals per slot and λ_j, as usual, the arrival rate of class j arrivals. Lastly, M denotes the number of classes the arrivals are categorized in. For $M = 1$ and $N = 1$ this distribution simplifies to a Bernoulli distribution and for $M = 1$ and $N > 1$ it simplifies to a binomial distribution.

The multinomial process is the input process to one of the output queues of an $N \times N$ output queueing switch as shown in Figure 1.7. The numbers of arrivals to the inlets are assumed to be i.i.d. and generated by a Bernoulli process with arrival rate λ_T. An arrival is of type j with probability $\frac{\lambda_j}{\lambda_T}$. Furthermore, the routing from inlets to outputs is uniform and independent, which makes the input to each of the N queues stochastically identical. The numbers of class-j arrivals to an output during a slot are mutually correlated. If n class-i arrivals enter the output queue in a slot, the maximum number of arrivals of any other class is limited to $N - n$ as only one arrival per inlet is possible.

We use this type of distribution frequently to test our results (i.e., compare calculations with simulations) and demonstrate the application of the presented theory. This however by no means limits the applicability of the methods to this kind of arrival process.

> For the simulations of the previous section, specifically Figures 1.4-1.6, we used a two-dimensional multinomial arrival process with parameters: $N = 16, \lambda_1 = 0.72$ and $\lambda_2 = 0.18$. Additionally, for Figures 1.4 and 1.5 we used $\beta = 0.5$.

1.7 State-of-the-art and related scheduling disciplines

Generalized Processor Sharing is a well-studied topic in the literature. In this section, we discuss the most important GPS studies related to this dissertation and some related scheduling disciplines. We start by referring the reader to the survey of Aalto et al. that discusses scheduler evolutions beyond (egalitarian) processor sharing [1].

In the original GPS paper [110], Parekh and Gallager prove worst-case delay and backlog bounds when the traffic is leaky-bucket constrained (which could for instance be imposed by a call admission control system). Analysis of GPS with exponentially bounded burstiness traffic has been carried out in [150,154]. Other work on call admission schemes for GPS is done in [48, 49, 115, 153].

A lot of GPS studies have been devoted to calculating the asymptotic tail behavior of the workload of a class [9, 20, 26, 27, 50, 97, 105, 151, 152]. This is for instance also the case in the PhD thesis of Miranda van Uitert [131] where the tail asymptotics are considered for heavy-tailed flows or a mixture of both heavy and light-tailed flows. Based on asymptotic results, the optimization of the GPS weights has been studied in [55, 93, 99–101]. For the optimization of multiclass, single-server queueing systems, conservation laws and achievable region methods are also widely used [11,17–19,60]. In this dissertation, we mainly focus on the mean performance measures and arrival processes without heavy tails. As a result, the conservations laws and achievable region methods in this work are of a different nature.

It has been shown that a two-class GPS system is equivalent to a coupled processor model, which can thus be solved analytically as a boundary value problem [41]. For the discrete-time implementation of this dissertation this has also been shown in the appendix of [145]. The extension to three classes, however, makes the numerical analysis very intricate [40]. We comment further on the challenges with this boundary value technique at the start of Part II on analysis of discrete-time GPS queues (See Section 4.1).

One can view our discrete-time probabilistic version of GPS as a practical implementation that solves the infinite divisibility assumption of continuous-time GPS. In the literature, other implementations have been studied, such as Weighted Fair Queueing (WFQ) also known as Packet-by-packet Generalized Processor Sharing (PGPS), Weighted Round Robin (WRR) and Discriminatory Round Robin (DRR). In Weighted Fair Queueing at each arrival instant, the scheduler calculates the virtual finish time of that customer in an equivalent

GPS system. At each departure the scheduler chooses the customer with the earliest virtual finish time as the next customer for service. In Weighted Round Robin the queues are visited in a round robin fashion; in proportion to their weight some queues are allowed to serve more than one customer. Lastly, Deficit Round Robin was used to solve the problem that classes with longer customer service times obtain a higher bandwidth relative to their weight in WRR. To this end a deficit counter was introduced and each time the queue is visited this counter is increased with that class its timeshare. Only when the service time of the next customer in that queue is smaller than the deficit counter, it gets served and the deficit counter is reduced with that service time. For single slot service times as mostly assumed in this dissertation, the distinction between WRR and DRR degenerates. Depending on the point of view, these scheduling mechanisms can be viewed as practical implementations of the ideal GPS or, the other way around, GPS can be used to study and approximate the performance of these schedulers. The exact analysis of these systems is typically harder. For more information, we refer the reader to [9,14,34,83,86,119,121].

Related to GPS is Discriminatory Processor Sharing (DPS). In DPS, like in GPS, customers are enqueued in class-specific queues and class i is assigned a certain weight ϕ_i. Unlike in GPS, where only the head-of-line customers of each queue are simultaneously served, in DPS, all customers are simultaneously served and a customer of class i is served at a fraction $\frac{\phi_i}{\sum_j \phi_j u_j}$ of the total outgoing bandwidth. In DPS, the bandwidth assigned to a class is thus depending on the queue contents of all classes. Clearly, DPS is also an idealized scheduling discipline that can be used as an approximation to practical schedulers. We refer the reader to the survey of Altman et al. [5] and some specific studies [12,59,129].

Lastly, we briefly mention that our discrete-time implementation of GPS could also be viewed as a discrete-time polling model with random server routing and 1-limited service. For a survey on polling models, we refer the reader to [28, 125, 140]. In continuous-time, a model with random server routing is for instance analyzed in [6] for a closed network. In [29], Boxma studies the optimization of the parameters for the random server routing to build an optimal static routing table for which the spacing between classes in the table is determined using [78]. Boxma and Weststrate [30] determine a pseudo-conservation law for a polling system with Markovian routing of the server. Kleinrock and Levy [90] calculated the mean response time of a random customer in a fully symmetric, discrete-time polling model with random server routing, switchover periods and 1-limited service. With fully symmetric, we mean symmetric in the arrival process, the switchover periods and the routing probabilities. This requirement in our case translates to a symmetric arrival process $A(z_1, z_2) = A(z_2, z_1)$ and $\beta = 0.5$. In the dissertation of van der Mei [127], polling systems with Bernoulli and Markovian server routing are studied. The author therefor uses a power series expansion in the total load offered to the system and uses that to optimize a linear cost function of the mean waiting times via the server routing parameters.

1.8 Outline of this dissertation

In this section, we present the structure of this dissertation. We also take advantage of this moment to reference our publications that also present the research from the corresponding chapter.

This dissertation is composed of two parts. In the first part, we study the achievable performance and the consequences for the optimization of the discrete-time implementation of Generalized Processor Sharing. In chapter 2, we study the two-class version we presented here. We start by proving that the achievable performance region is bounded by the two strict priority boundary cases. Furthermore, we show that the mean unfinished work of the classes is continuous and monotone as a function of β [146]. Using these important properties, we study the optimization of general objective functions, and some more specific ones [133,138]. For instance, we prove that if the objective function is a weighted combination of functions g_j of the mean unfinished work of the classes, that pure GPS is only optimal if the g_j are convex increasing. If the g_j are concave or linearly increasing, strict priority is always optimal. We end the chapter by proving that the applicability of the most important theorems extends to alternative scheduling mechanisms, other than GPS [134].

In chapter 3, we study the extension from our two-class model to a three-class model. We propose to use a hierarchical version, that has advantageous properties for the optimization and prove that it has the same achievable performance region as the non-hierarchical version [137]. We also present an algorithm that calculates the configuration of the scheduler to obtain a desired performance. The chapter is concluded by discussing the extension to more than three classes.

In the second part, we focus on the analysis of the system. In chapter 4, we start by summarizing a power series expansion of the joint pgf of the unfinished work for the two-class system. This power series is no work of the author of this dissertation but is found in the literature. Our contribution here is to study the challenges inherent to the expansion that limit the number of calculable coefficients in the power series. We propose some techniques to get the most out of the approximation and evaluate the result by applying it to an optimization problem [135].

In chapter 5, we extend the power series expansion to the three-class hierarchical version from chapter 3. We also study a special case where there is one high-priority class and the remainder of the capacity is shared using GPS amongst the other two classes [136].

In chapter 6, we take a step back to find a method that makes it possible to calculate more coefficients in the power series of the mean unfinished work. We do that by directly calculating the means from the system equations. This eliminates the problems that are associated with the detour via the pgf. The method is developed for the two-class system serving a simpler arrival process, than in the rest of the dissertation. This simplifies the analysis somewhat, but does not simplify the inherent difficulties from the GPS system.

In chapter 7, we summarize the most important conclusions of this dissertation.

Part I

Achievable Region and Optimization

"Wat gij niet wilt dat u geschiedt, doe dat ook een ander niet."

— Esther Roelens

2 Two Classes

2.1 Achievable Region

In this chapter, we study the optimization of the two-class model described in Section 1.3. In this section, we start by proving which performances are achievable and unachievable. The achievable region of this model has a great influence on the optimization thereof, which is investigated in the next section. Lastly, in the last section of this chapter, we show that these results can be generalized to a wider class of models, not limited to GPS.

To recap, we study the two-class, discrete-time GPS model from Section 1.3. The arrivals are generated by a general arrival process that is independent from slot to slot. Each arrival requires a single slot of service. Furthermore, we assume that the stability condition $\lambda_T < 1$ is met, so that the system reaches a steady state in the long term. The analysis of these kinds of GPS queueing models is notoriously hard, which is why we look for other means of obtaining important properties of the performance measures to subsequently use in optimization.

Studying the performance of the system, we focus on the mean unfinished work of the individual classes in the system. As each customer requires a single slot of service, the mean unfinished work equals the mean queue length. It is clear that $\bar{w}_1$ and $\bar{w}_2$ are functions of β, the scheduling parameter, i.e., $\bar{w}_1 = \bar{w}_1(\beta)$ and $\bar{w}_2 = \bar{w}_2(\beta)$.

With $\beta = 0$, we have already shown (using the definition of the model) that class-1 customers have the lowest priority in the system. Consequently, $\bar{w}_1(0) = \bar{w}_{1,\max}$ as it is impossible with $\beta \in [0, 1]$ and keeping the system work-conserving to keep more class-1 work in the system, than only serving it when no other work is present, which is the case when $\beta = 0$. Conversely, $\beta = 1$ gives $\bar{w}_1(1) = \bar{w}_{1,\min}$, as this represents strict priority for class-1 customers. Intuitively one feels that $\bar{w}_1(\beta)$ decreases when β increases. The aim of this section is to prove that $\bar{w}_1(\beta)$ is a strictly decreasing and continuous function. This means that for every value $\bar{w}_1^*$ in the interval $[\bar{w}_{1,\min}, \bar{w}_{1,\max}]$ there exist a unique β^* such that $\bar{w}_1^* = \bar{w}_1(\beta^*)$.

Before we formulate the theorem, we first define regenerative processes and regeneration cycles as it is required in the theorem. A process is regenerative when the process can be split into independent and identically distributed regeneration cycles [7]. As we assume that $\lambda_T < 1$ (stability condition), the probability that the system is empty and the server idle is non-zero. It is easily seen that the *busy cyles*, i.e. the cyles where the server is non-idle or busy, are such regeneration cycles of the system[1]. At the start of each busy cycle the system starts anew indepedently from the past and governed by the same probability law.

Theorem 2.1. *The function* $\bar{w}_2(\beta)$ *(*$\bar{w}_1(\beta)$*) is a strictly increasing (decreasing) function on the interval* $[0, 1]$*, if* $\Pr[w_1(\beta) > 0, w_2(\beta) > 0] > 0$*. Furthermore, both functions are continuous on the same interval if* $\mathrm{E}[T^2]$ *is finite, with* T *the length of a random regeneration cycle.*

For the proof of Theorem 2.1, we require some lemmas that we will handle first. We postpone the presentation of the proof and the explanation of the extra assumptions. In that proof, we couple two systems in such a way that for equal β their sample paths are equal from slot to slot. To this end all input processes are coupled, i.e. arrivals in each slot are identical in both systems. To couple the decision making processes of the schedulers, we formalize this process first. Each slot the scheduler generates a decision variable, the decision variable in slot k is denoted by r_k and is drawn from a uniform distribution on $[0, 1]$. In case there is contention between both classes, i.e. customers of both classes are present, the scheduler serves class 1 when $r_k \in [0, \beta]$ and class 2 when $r_k \in]\beta, 1]$. As such, class 1 (class 2) is served with probability β $(1 - \beta)$ conform to the specification of the policy.

[1] It is possible to identify other regeneration cycles. The splitting in regeneration cycles is not unique.

Perturbation analysis

The method we use here is known as perturbation analysis. In perturbation analysis, a single sample path is analyzed. The generation of a perturbation by changing a single system parameter (for instance changing the weight parameter from β to $\beta + \Delta\beta$) propagates a series of perturbations on the sample path. These perturbations of the sample path result in changes of the system performance, which is the topic of the study. In essence the perturbation analysis is a sensitivity analysis of a parameter on the system performance.

In this section the perturbation analysis is used to show that the increase of the weight parameter (β) leads to a bounded increase of the mean unfinished work of class 2. Which enables us to prove monotonicity and continuity of the mean unfinished work. These results in turn aid to prove more complex results on the optimization of the system in the following sections. In Part II, we will anew turn to perturbation analysis for deriving approximations for the performance characteristics. We will come back to the specific techniques used there at the appropriate time. For a general description of perturbation techniques we refer to [149].

For the proof of Theorem 2.1, we couple two systems with a different weight parameter and study the influence thereof. In particular, we couple a β- and a $\beta + \Delta\beta$-system, with $0 \leq \beta < \beta + \Delta\beta \leq 1$. The following two lemmas compare the sample paths of these systems.

Lemma 2.1. *The total amount of unfinished work in the two systems, coupled as explained above, are equal in every slot.*

Proof. The number of arrivals is identical in both of the coupled systems. As both systems are work-conserving, they have to serve a customer whenever work is present at the beginning of the slot. Consequently, both systems serve customers (albeit possibly from a different class) and idle in the same slots. As a result, in every slot both systems have the same amount of unfinished work. □

In Lemma 2.1, we basically prove that the total unfinished work is independent of β, i.e. independent of the order in which customers are served. In symbols, we write $w_{1,k}(\beta) + w_{2,k}(\beta) = w_{T,k}$, whereby we have explicitly denoted dependence of β where appropriate and $w_{T,k}$ denotes the total unfinished work in the system at the beginning of slot k.

To easily study the difference between the two systems, we define $\Delta w_{j,k}(\beta) \triangleq w_{j,k}(\beta + \Delta\beta) - w_{j,k}(\beta)$ and $\Delta\bar{w}_j(\beta) \triangleq \mathrm{E}[\Delta w_{j,k}(\beta)]$. In the following lemma, we uncover in what situations differences between the β- and $\beta + \Delta\beta$-system change by comparing all possibilities from slot k to the next. In this lemma, we use the operator $(\cdot)^+$ defined as $\max(\cdot, 0)$. Furthermore, we only concentrate on $w_{2,k}$ as $w_{1,k}$ can be calculated from $w_{T,k}$ per Lemma 2.1.

Lemma 2.2. *The following inequalities hold for all k:*

(i) $\Delta w_{2,k}(\beta) \geq 0$,

(ii) $(\Delta w_{2,k}(\beta) - 1)^+ \leq \Delta w_{2,k+1}(\beta) \leq \Delta w_{2,k}(\beta) + 1$, *if* $r_k \in]\beta, \beta + \Delta\beta]$, *and*

(iii) $(\Delta w_{2,k}(\beta) - 1)^+ \leq \Delta w_{2,k+1}(\beta) \leq \Delta w_{2,k}(\beta)$, *if* $r_k \notin]\beta, \beta + \Delta\beta]$.

Proof. We first prove (ii) and (iii) conditionally on $\Delta w_{2,k}(\beta) \geq 0$. Later, we will prove (i), which also means that this condition is always met.

We start with the situation where $\Delta w_{2,k}(\beta) = 0$. If $r_k \notin]\beta, \beta + \Delta\beta]$, the same decision is taken for both systems in slot k, leading to $\Delta w_{2,k+1}(\beta) = 0$. If $r_k \in]\beta, \beta + \Delta\beta]$, we distinguish four cases:

1. $w_{1,k} = 0, w_{2,k} = 0$,
2. $w_{1,k} = 0, w_{2,k} > 0$,
3. $w_{1,k} > 0, w_{2,k} = 0$,
4. $w_{1,k} > 0, w_{2,k} > 0$.

In the first three cases, the same (or no) work unit is served in slot k in both systems, yielding $\Delta w_{2,k+1}(\beta) = \Delta w_{2,k}(\beta)$. In case 4, a work unit of class 2 is served in the β-system, while a work unit of class 1 is served in the $\beta + \Delta\beta$-system, leading to $\Delta w_{2,k+1}(\beta) = \Delta w_{2,k}(\beta) + 1$. It is thus clear that (ii) and (iii) hold if $\Delta w_{2,k}(\beta) = 0$.

Let us next turn to the case $\Delta w_{2,k}(\beta) > 0$. Evidently, $w_{2,k}(\beta + \Delta\beta)$ and $w_{1,k}(\beta)$ are then strictly positive ($\Delta w_{1,k}(\beta) = -\Delta w_{2,k}(\beta)$, cf. Lemma 2.1). The situation in this slot can be one of ten:

1. $w_{1,k}(\beta + \Delta\beta) = w_{2,k}(\beta) = 0$,
2. $w_{1,k}(\beta + \Delta\beta) = 0, w_{2,k}(\beta) > 0, r_k \leq \beta$,
3. $w_{1,k}(\beta + \Delta\beta) = 0, w_{2,k}(\beta) > 0, \beta < r_k \leq \beta + \Delta\beta$,
4. $w_{1,k}(\beta + \Delta\beta) = 0, w_{2,k}(\beta) > 0, \beta + \Delta\beta < r_k$,
5. $w_{1,k}(\beta + \Delta\beta) > 0, w_{2,k}(\beta) = 0, r_k \leq \beta$,
6. $w_{1,k}(\beta + \Delta\beta) > 0, w_{2,k}(\beta) = 0, \beta < r_k \leq \beta + \Delta\beta$,
7. $w_{1,k}(\beta + \Delta\beta) > 0, w_{2,k}(\beta) = 0, \beta + \Delta\beta < r_k$,
8. $w_{1,k}(\beta + \Delta\beta) > 0, w_{2,k}(\beta) > 0, r_k \leq \beta$,

9. $w_{1,k}(\beta + \Delta\beta) > 0, w_{2,k}(\beta) > 0, \beta < r_k \leq \beta + \Delta\beta$,
10. $w_{1,k}(\beta + \Delta\beta) > 0, w_{2,k}(\beta) > 0, \beta + \Delta\beta < r_k$.

According to the scheduling rules, the following services take place in slot k:

- in cases 5-6 and 8, a class-1 customer service in both systems,
- in cases 3-4 and 10, a class-2 customer service in both systems,
- in cases 1-2 and 7, a class-1 customer service in the β-system and a class-2 customer service in the $\beta + \Delta\beta$-system, and,
- in case 9, a class-2 customer service in the β-system and a class-1 customer service in the $\beta + \Delta\beta$-system.

Propositions (ii) and (iii) follow immediately: $\Delta w_{2,k}(\beta) - 1 \leq \Delta w_{2,k+1}(\beta) \leq \Delta w_{2,k}(\beta)$ in all cases except case 9. In the latter case, $\beta < r_k \leq \beta + \Delta\beta$ and $\Delta w_{2,k+1}(\beta) = \Delta w_{2,k}(\beta) + 1$.

Finally, we prove (i) by induction on k. It follows from the first part of this proof that $\Delta w_{2,k+1}(\beta) \geq (\Delta w_{2,k}(\beta) - 1)^+$ and therefore $\Delta w_{2,k+1}(\beta) \geq 0$. The inductive proof is concluded by the coupling assumption on the initial unfinished work, i.e., by $\Delta w_{2,1}(\beta) = 0$. $\square$

With these lemmas we are ready to formulate the proof of Theorem 2.1. In the first part of the proof, we establish a lower bound on $\Delta\bar{w}_2(\beta)$, thus proving strict monotonicity of $\bar{w}_2(\beta)$. The main step in this part of the proof uses the system dynamics from Lemma 2.2. In the second part, we establish an upper bound on $\Delta\bar{w}_2(\beta)$ to prove continuity. This part uses Lemma 2.2 but also requires the introduction of the extra assumptions on the distribution of the length of the regeneration cycle.

Proof of Theorem 2.1. We focus on the effect of a positive displacement $\Delta\beta$ on $\Delta\bar{w}_2(\beta)$ and prove that $0 < \lim_{\Delta\beta\to 0} \frac{\Delta\bar{w}_2(\beta)}{\Delta\beta} < \infty$ (under the conditions stated in the theorem). The same reasoning holds for a negative displacement $-\Delta\beta$. We can then conclude that $\bar{w}_2(\beta)$ is a continuous, strict monotonic function on the interval $[0, 1]$.

We pick a random slot and denote it by slot J. We will calculate lower (to prove strict monotonicity) and upper (for continuity) bounds on the expected difference between the class-2 unfinished work at the beginning of that slot in both coupled β- and $\beta + \Delta\beta$-systems.

We start with the lower bound. We can write

$$\Delta\bar{w}_2(\beta) \geq \mathrm{E}[\Delta w_{2,J}(\beta)\mathbb{1}_{w_{1,I}(\beta)>0, w_{2,I}(\beta)>0}],$$

with slot I the slot preceding slot J. From the proof of Lemma 2.2, it follows that under the condition in the previous equation, the difference in $\Delta w_2(\beta)$ increases with 1 from slot I to slot J if the decision variable r_I is in $]\beta, \beta + \Delta\beta]$ and stays the same if r_I is not in this interval, i.e.,

$$\begin{aligned}\Delta\bar{w}_2(\beta) \geq & \mathrm{E}[(\Delta w_{2,I}(\beta) + 1)\mathbb{1}_{w_{1,I}(\beta)>0, w_{2,I}(\beta)>0}]\Delta\beta \\ & + \mathrm{E}[\Delta w_{2,I}(\beta)\mathbb{1}_{w_{1,I}(\beta)>0, w_{2,I}(\beta)>0}](1 - \Delta\beta).\end{aligned}$$

By using (i) of Lemma 2.2 and the fact that slot I is a random slot, we can further write

$$\Delta\bar{w}_2(\beta) \geq \Pr[w_1(\beta) > 0, w_2(\beta) > 0]\Delta\beta.$$

Under the assumption of the theorem, $\Pr[w_1(\beta) > 0, w_2(\beta) > 0] > 0$, independently of $\Delta\beta$, and, therefore $\Delta\bar{w}_2(\beta)$ is strictly positive and, consequently, $\bar{w}_2(\beta)$ strictly increasing.

We now establish an upper bound to prove continuity. Since the difference is 0 at the beginning of the regeneration cycle slot J is part of, it will be of interest to characterize the age of the on-going regeneration cycle. Therefore, introduce $\tilde{T}$ as the length of the elapsed part of the on-going regeneration cycle at the beginning of slot J and $\tilde{r}(\beta, \Delta\beta)$ as the number of times the decision variable was in $]\beta, \beta + \Delta\beta]$ during this elapsed part. Lemma 2.2, part (ii) then states that in each of the slots the decision variable is in $]\beta, \beta + \Delta\beta]$ the difference in the unfinished work of class 2 can at most increase with 1. From part (iii) of that lemma, it follows that in the other slots a further increase is not possible. The difference in unfinished work of class-2 can therefore be at maximum $\tilde{r}(\beta, \Delta\beta)$ and we have

$$\Delta\bar{w}_2(\beta) \leq \mathrm{E}[\tilde{r}(\beta, \Delta\beta)]. \tag{2.1}$$

We further bound the right-hand side. Due to the law of total probability, we can write

$$\mathrm{E}[\tilde{r}(\beta, \Delta\beta)] = \sum_{k=0}^{\infty} \mathrm{E}[\tilde{r}(\beta, \Delta\beta)|\tilde{T} = k] \Pr[\tilde{T} = k].$$

Since I is a randomly chosen slot, the inspection paradox yields

$$\mathrm{E}[\tilde{r}(\beta, \Delta\beta)] = \sum_{k=0}^{\infty} \mathrm{E}[\tilde{r}(\beta, \Delta\beta)|\tilde{T} = k] \sum_{l=k+1}^{\infty} \frac{\Pr[T = l]}{\mathrm{E}[T]}, \tag{2.2}$$

with T the length of a random regeneration cycle. Since the server allocation variables are independent and uniformly distributed in $[0, 1]$, they are in the interval $]\beta, \beta+\Delta\beta]$ with probability $\Delta\beta$, and $\tilde{r}(\beta, \Delta\beta)$ has a binomial distribution with parameters $\tilde{T}$ and $\Delta\beta$. We obtain

$$\begin{aligned} \mathrm{E}[\tilde{r}(\beta, \Delta\beta)|\tilde{T} = k] &= \sum_{l=0}^{k} \binom{k}{l} \Delta\beta^l \cdot (1-\Delta\beta)^{k-l} \cdot l \\ &= k \cdot \Delta\beta \end{aligned} \tag{2.3}$$

Substitution of (2.3) in (2.2) leads to

$$\begin{aligned} \mathrm{E}[\tilde{r}(\beta, \Delta\beta)] &= \sum_{k=0}^{\infty} k \cdot \Delta\beta \sum_{l=k+1}^{\infty} \frac{\Pr[T=l]}{\mathrm{E}[T]} \\ &= \frac{\Delta\beta}{\mathrm{E}[T]} \sum_{l=1}^{\infty} \Pr[T=l] \left(\sum_{k=0}^{l-1} k \right) \\ &= \frac{\Delta\beta}{\mathrm{E}[T]} \sum_{l=1}^{\infty} \Pr[T=l] \frac{(l-1)l}{2} \\ &\leq \frac{\mathrm{E}[T^2]}{2\mathrm{E}[T]} \Delta\beta. \end{aligned}$$

Using this in (2.1), we find that, for each $\Delta\beta$,

$$\Delta\bar{w}_2(\beta) \leq \frac{\mathrm{E}[T^2]}{2\mathrm{E}[T]} \Delta\beta.$$

Taking the limit $\Delta\beta \to 0$ leads to the theorem. □

Infinite second moment of regeneration cycle length

We see that to prove continuity the extra requirement of a finite second moment for the length of the regeneration cycle arises. As a corollary, if this condition is not fulfilled it is possible for the mean unfinished work of a class to be discontinuous for a certain value of β. In fact, based on the proof it is possible to construct such an example. We present one here.

Assume an arrival process that is arranged in such a way that in the first slot of a regeneration cycle there is always the arrival of exactly one class-1 customer and at least one class-2 customer. In the remaining slots of the regeneration cycle, there are only class-2 arrivals in such a way that the total number of class-2 arrivals in the elapsed part of the regeneration cycle is always larger than the length of the elapsed part. Then in case $\beta = 0$, we have strict priority for class 2 and the class-1 customer that arrives at the start of the regeneration cycle will always be the last customer to get served. Introducing our perturbation, we look at $\beta = \Delta\beta$. From the proof of Theorem 2.1, we have that (2.1), for this specific case, changes to

$$\Delta\bar{w}_2(0) = \mathrm{E}[\hat{r}(0, \Delta\beta)],$$

whereby $\hat{r}(0, \Delta\beta)$ is defined to be 0 if none of the decision variables lies in $]0, \Delta\beta]$ during $\tilde{T}$ and 1 if there is at least one. Indeed, if $\hat{r}(0, \Delta\beta) = 0$ then $\Delta w_{2,J}(\beta) = 0$, there will be no difference, the class-1 customer is in both systems still waiting. However, if $\hat{r}(0, \Delta\beta) = 1$ then $\Delta w_{2,J}(\beta) = 1$, the class-1 customer is already served in the $0 + \Delta\beta$-system. Alternatively, we can define $\hat{r}(0, \Delta\beta) \triangleq \min(\tilde{r}(0, \Delta\beta), 1)$.

We can then continue the analysis in a similar way as in the proof.

$$\begin{aligned}
\mathrm{E}[\hat{r}(0, \Delta\beta)] &= \sum_{k=0}^{\infty} \mathrm{E}[\hat{r}(0, \Delta\beta)|\tilde{T} = k] \Pr[\tilde{T} = k] \\
&= \sum_{k=0}^{\infty} \mathrm{E}[\hat{r}(0, \Delta\beta)|\tilde{T} = k] \sum_{l=k+1}^{\infty} \frac{\Pr[T = l]}{\mathrm{E}[T]}
\end{aligned}$$

From the definition of $\hat{r}(0, \Delta\beta)$, we can see that $\mathrm{E}[\hat{r}(0, \Delta\beta)|\tilde{T} = k]$ equals the probability that at least one decision variable in the k preceding slots of the busy cycle is in $]0, \beta + \Delta\beta]$. Equivalently, this is 1 minus the probability that no decision variable is in that interval. We thus find: $\mathrm{E}[\hat{r}(0, \Delta\beta)|\tilde{T} = k] = 1 - (1 - \Delta\beta)^k$. We use this in our previous equations and develop the expressions further:

$$\begin{aligned}
\Delta\bar{w}_2(0) &= \sum_{k=0}^{\infty} \Big(1 - (1 - \Delta\beta)^k\Big) \sum_{l=k+1}^{\infty} \frac{\Pr[T = l]}{\mathrm{E}[T]} \\
&= \frac{1}{\mathrm{E}[T]} \sum_{l=1}^{\infty} \Pr[T = l] \sum_{k=0}^{l-1} \Big(1 - (1 - \Delta\beta)^k\Big) \\
&= \frac{1}{\mathrm{E}[T]} \sum_{l=1}^{\infty} \Pr[T = l] \Big(l - \frac{1 - (1 - \Delta\beta)^l}{\Delta\beta}\Big) \\
&= 1 - \frac{1}{\mathrm{E}[T]\Delta\beta} \sum_{l=1}^{\infty} \Pr[T = l] \Big(1 - (1 - \Delta\beta)^l\Big) \\
&= 1 - \frac{1}{\mathrm{E}[T]\Delta\beta} + \frac{1}{\mathrm{E}[T]\Delta\beta} \sum_{l=1}^{\infty} \Pr[T = l](1 - \Delta\beta)^l \\
&= \frac{\mathrm{E}[T]\Delta\beta - 1 + T^*(1 - \Delta\beta)}{\mathrm{E}[T]\Delta\beta},
\end{aligned}$$

whereby we have used that $\Pr[T = 0] = 0$ and defined $T^*(z)$ as the probability generating function of the regeneration cycle length, i.e., $T^*(z) = \mathrm{E}[z^T] = \sum_{l=1}^{\infty} z^l \Pr[T = l]$.

We then calculate the limit:

$$\begin{aligned}\lim_{\Delta\beta\to 0} \frac{\Delta\bar{w}_2(0)}{\Delta\beta} &= \lim_{\Delta\beta\to 0} \frac{\mathrm{E}[T]\Delta\beta - 1 + T^*(1-\Delta\beta)}{\mathrm{E}[T]\Delta\beta^2} \\ &\overset{\mathrm{H}}{\underset{\frac{0}{0}}{=}} \lim_{\Delta\beta\to 0} \frac{\mathrm{E}[T] + \mathcal{D}(T^*)(1-\Delta\beta)}{2\mathrm{E}[T]\Delta\beta} \\ &\overset{\mathrm{H}}{\underset{\frac{0}{0}}{=}} \lim_{\Delta\beta\to 0} \frac{\mathcal{D}^2(T^*)(1-\Delta\beta)}{2\mathrm{E}[T]} \\ &= \frac{\mathrm{E}[T^2] - \mathrm{E}[T]}{2\mathrm{E}[T]}\end{aligned}$$

In these calculations, we have used l'Hopital's rule and the moment generating property of the probability generating function, in particular, $T^*(1) = 1, \mathcal{D}(T^*)(1) = \mathrm{E}[T]$ and $\mathcal{D}^2(T^*)(1) = \mathrm{E}[T^2] - \mathrm{E}[T]$. We find that if $\mathrm{E}[T^2]$ the second moment of the regeneration cycle length is infinite, the limit does not exist and $\bar{w}(\beta)$ is discontinuous in $\beta = 0$. This shows the necessity of the conditions on the probability distribution of the regeneration cycle length in Theorem 2.1.

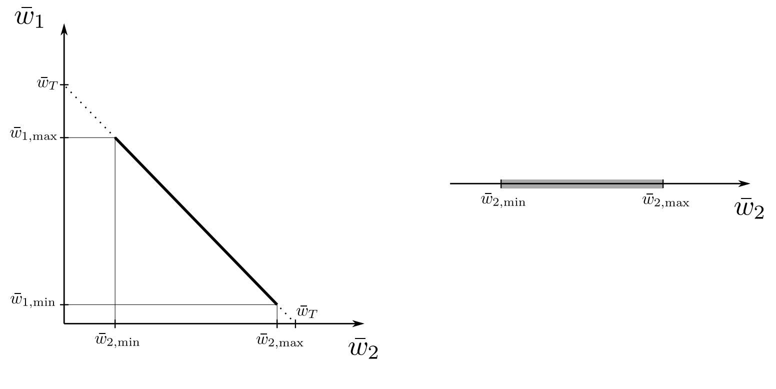
Extension

The results from this section, are valid for a wider class of models. In particular, they are also valid if interruptions of the server are possible (meaning the server can either be serving customers in a slot or be offline). Moreover, they are also valid for customers that require longer service times when they are served in a preemptive manner. This means that at every slot boundary the service of a customer can be paused to start/continue serving a customer of another class. Afterwards the service of the paused customer is simply resumed from where it stopped. This extended result is described in [146].

Summary

The two most important results from this section to remember for the remainder of this dissertation are delineating the achievable region of two-class discrete-time GPS. The first one is a corollary of Lemma 2.1.

Corollary 2.1. *The function $\bar{w}_1(\beta)+\bar{w}_2(\beta)$ is independent of β, i.e., equals $\bar{w}_T$.*



(a) (b)

Figure 2.1: Achievable region for two-classes GPS

$\bar{w}_T$ is the mean of the total unfinished work in the system and is a constant that can be calculated from single-class systems [32]. The second one is Theorem 2.1 describing the continuity and increasing, respectively decreasing, behavior of $\bar{w}_2(\beta)$ and $\bar{w}_1(\beta)$ with respect to β. As a result $\bar{w}_2(0) = \bar{w}_{2,\min}$ ($\bar{w}_1(1) = \bar{w}_{1,\min}$) the smallest possible value for $\bar{w}_2$ ($\bar{w}_1$) and $\bar{w}_2(1) = \bar{w}_{2,\max}$ ($\bar{w}_1(0) = \bar{w}_{1,\max}$) the largest possible value for $\bar{w}_2$ ($\bar{w}_1$). These cases of $\beta = 0$ or $\beta = 1$ correspond to strict priority for one of the classes. The values $\bar{w}_{1,\min}, \bar{w}_{1,\max}, \bar{w}_{2,\min}, \bar{w}_{2,\max}$ can easily be calculated using results from [143]. As a result from continuity, we can reformulate Bolzano's theorem also known as the intermediate value theorem for this model in the following corollary:

Corollary 2.2. *For every $\bar{w}_2^* \in [\bar{w}_2(0), \bar{w}_2(1)]$ there exists a unique $\beta^* \in [0,1]$ so that $\bar{w}_2(\beta^*) = \bar{w}_2^*$. Equivalently for class 1: for every $\bar{w}_1^* \in [\bar{w}_1(1), \bar{w}_1(0)]$ there exists a $\beta^* \in [0,1]$ so that $\bar{w}_1(\beta^*) = \bar{w}_1^*$.*

In Figure 2.1a, we have drawn the achievable region as described by Corollary 2.1 and 2.2. As can be seen it is a straight line, and only the performance vectors $\bar{\boldsymbol{w}} = (\bar{w}_1, \bar{w}_2)$ on this line are achievable with the system under study. The achievable region is thus one-dimensional and if we consider $\bar{w}_1$ ($= \bar{w}_T - \bar{w}_2$) as a dependent variable, we get the alternative description of the achievable region from Figure 2.1b. We have chosen $\bar{w}_2$ as the *independent* variable, as it increases with β which is easier to reason with in the rest of this chapter. We denote the set of achievable performance vectors by $\Omega \subset \mathbb{R}^2$. Furthermore, we define $\Omega_1 = [\bar{w}_1(1), \bar{w}_1(0)]$ and $\Omega_2 = [\bar{w}_2(0), \bar{w}_2(1)]$.

2.2 Optimization

In this section, we focus on the optimization of the two-class GPS model. A first task for optimization is to determine what you want to optimize. The optimization objective is formalized in a so-called objective function, which is a function of performance metrics of the system under optimization. Often this function is also called a cost function, in this setting the function expresses penalties associated with the performance metrics. The goal of optimization is then to minimize the cost function. In its generality an objective function does not always have to be minimized as it could also represent the utility of the performance metrics. In that case one would want to find the maximum. It is clear that in the terminology of a cost function one always aims at minimizing the cost.

The objective functions studied in this section are functions of the mean unfinished work of both classes. It is important to note that by wrapping the mean unfinished work of a class by an appropriate function, one can easily switch to an objective function in terms of another performance metric. For single slot service times as used in this dissertation the mean unfinished work of a queue equals the mean queue content. By applying Little's law, we can further transform to mean customer delay:

$$\bar{d}_j(\beta) = \frac{\bar{w}_j(\beta)}{\lambda_j}.$$

Various results from this dissertation easily extend to geometrically distributed service times with parameter μ_j for class j, combined with a preemptive service discipline[2]. In that case, we transform to the mean queue content $\bar{u}_j(\beta)$ as follows

$$\bar{u}_j(\beta) = \mu_j \bar{w}_j(\beta).$$

The mean delay than becomes

$$\bar{d}_j(\beta) = \frac{\mu_j \bar{w}_j(\beta)}{\lambda_j}.$$

So say, we want an objective function that is a function of the mean customer delay of both classes, i.e., $f\Big(\bar{d}_1(\beta), \bar{d}_2(\beta)\Big)$. In that case, we can easily transform it to a function of the mean unfinished work:

[2] In a preemptive service discipline the service of a customer can be interrupted (or pre-empted) at each slot boundary. The service of that customer is resumed from where it was interrupted later on.

$f\Big(h_1(\bar{w}_1(\beta)), h_2(\bar{w}_2(\beta))\Big)$ with $h_j(x) = \dfrac{\mu_j x}{\lambda_j}$. For simplicity, we therefore limit ourselves to studying objective functions in the mean unfinished work of both classes. Additionally, using the mean unfinished work enables us to use the important theorems from the previous section, whereas these theorems are not necessarily fulfilled for the other performance measures.

During the selection of an objective function it is good for the network operator to have some notion on the influence of her/his choices on the optimum. In a first step, the operator chooses the relevant performance characteristics and the type of relation of each performance characteristic in the objective function. When equal increments for high or low values should have an equal influence on the objective function, a linear relation can be used. The behavior of other types of relations can easily be derived from a plot of the corresponding function. Other examples are a squared relation or a logistic one. This choice of relation is closely related to the relation between Quality of Service (QoS) and Quality of Experience (QoE) [63,130] and the choice of utility functions [96,106]. A second question for the operator is how to weigh the performance of both classes. An answer to this question is less clear and in most cases more arbitrarily chosen.

In the first subsection, we look at a framework to study the optimum of general objective functions. In subsection 2.2.2, we study a subclass of objective functions, namely a weighted sum of increasing functions of the mean unfinished work of both classes. For this subclass, we analytically prove in what cases a minimum that occurs for different β than $\beta = 0$ or $\beta = 1$ exists. Subsequently in 2.2.3, we extend the subclass of objective functions from the preceding subsection and apply the framework from 2.2.1. This enables us to perform a sensitivity analysis on the weights used in the objective function, enabling an operator to assess the impact on the optimum and behavior of the objective function. Lastly in subsection 2.2.4, we apply the knowledge of the preceding subsections to form an efficient procedure to find a value for β that achieves the desired extremum of the objective function.

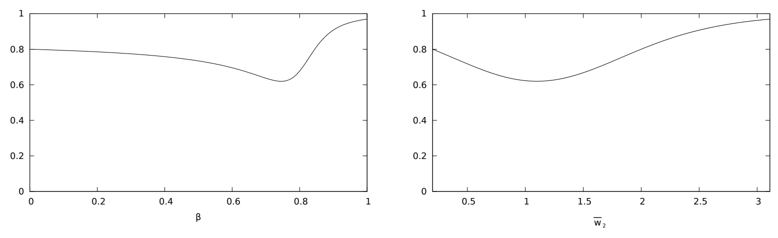
2.2.1 General Framework

For the optimization of single-variable (in this case β) functions various standard mathematical techniques have been developed. However, these are useless as we do not have expressions for the relation between the performance metrics used in the objective function and the optimization parameter β. Say, we have an objective function that is a generic function of the mean unfinished work in both queues, i.e., $f(\bar{w}_1(\beta), \bar{w}_2(\beta))$. From the optimization point of view, this is a function in terms of β: $F(\beta) \triangleq f(\bar{w}_1(\beta), \bar{w}_2(\beta))$. The standard technique of finding the critical points[3] and subsequently going from there, is unusable here. This is because we do not have analytical solutions for $\bar{w}_j(\beta)$. The analysis of GPS queueing systems is hard, even for the simplest arrival process, except for $\beta = 0$ or $\beta = 1$ which are strict priority cases. In Part II, we obtain some approximations for these solutions that can aid in optimization. Nevertheless, in this part of the dissertation we ignore these approximations and develop some exact analytical results regarding optimization.

The main observation is that the objective function $f(\bar{w}_1(\beta), \bar{w}_2(\beta))$, can also be expressed in terms of a single variable, other than β. In particular, Corollary 2.1 states that $\bar{w}_1(\beta) = \bar{w}_T - \bar{w}_2(\beta)$, implying that $f(\bar{w}_1(\beta), \bar{w}_2(\beta))$ can be expressed in terms of $\bar{w}_2(\beta)$ only. With a slight abuse of notation, we define this form as $f^*(\bar{w}_2)$. It is obvious that $F = f^* \circ \bar{w}_2$. With missing expressions for $\bar{w}_2(\beta)$ the study of F requires estimations/approximations for this performance measure. However, since we know the image ($\Omega_2 = [\bar{w}_2(0), \bar{w}_2(1)]$) and behavior (continuous and strictly increasing) of $\bar{w}_2(\beta)$, we can already study $f(\bar{w}_1(\beta), \bar{w}_2(\beta))$ in terms of $\bar{w}_2(\beta)$ instead of β, with domain Ω_2 instead of $[0, 1]$ (i.e., studying $f^*(\bar{w}_2)$).

Consequently, we can observe the number of extrema and inflection points and determine the values of $f(\bar{w}_1(\beta), \bar{w}_2(\beta))$ in these points without running simulations or relying on possibly inaccurate approximate expressions. Obviously, we do not know the β-values corresponding to these points (except when they coincide with the endpoints). To determine these β-values, we still need estimations/approximations. However, some important preliminary conclusions can be drawn from the behavior of $f^*(\bar{w}_2)$. For instance, the minimum could be in one of the endpoints $\beta = 0$ or $\beta = 1$. In that case, strict priority is optimal and we do not have to simulate. Another possible conclusion is that the difference in the objective function between the minimum and one of the endpoints is too small to justify a time-consuming quest for the β-value corresponding to the minimum. Summarized, from the analysis of $f^*(\bar{w}_2)$, an interval in Ω_2 with an acceptable value for the objective function can be selected. The optimization problem then reduces to finding a value of β for which the continuous and monotonic function $\bar{w}_2(\beta)$ reaches a

[3]Critical points are the points with $F'(\beta) = 0$; these are the only candidates to be an extremum.



(a) $f_1(\bar{w}_1(\beta), \bar{w}_2(\beta))$ (b) $f_1^*(\bar{w}_2)$

Figure 2.2: Comparison between the objective function $f_1(\bar{w}_1(\beta), \bar{w}_2(\beta))$ and $f_1^*(\bar{w}_2)$

value in the selected interval (stopping criterion). We demonstrate these findings by means of some illustrative examples.

Some Illustrative Examples

For the examples, we generated the number of class-1 and class-2 arrivals by a two-dimensional binomial distribution as introduced in Section 1.6. Adopting some concrete values for the parameters of the arrival process, the priority results of [143] can be used to calculate $\bar{w}_j(0)$ and $\bar{w}_j(1)$ ($j = 1, 2$). For $N = 16$, $\lambda_T = 0.9$, and $\alpha = 0.8$, for example, we find that

$$\begin{aligned} \bar{w}_1(0) &= 4.50, \qquad \bar{w}_2(0) = 0.20, \\ \bar{w}_1(1) &= 1.59, \qquad \bar{w}_2(1) = 3.11. \end{aligned} \tag{2.4}$$

Then $\bar{w}_T = \bar{w}_1(\cdot) + \bar{w}_2(\cdot) = 4.70$ and the intervals Ω_1 and Ω_2 are given by $[1.59, 4.5]$ and $[0.2, 3.11]$, respectively.

As objective function for our first example, we use two logistic functions:

$$f_1(\bar{w}_1(\beta), \bar{w}_2(\beta)) \triangleq \frac{1}{1 + e^{-2\bar{w}_1(\beta)+8}} + \frac{1}{1 + e^{-2\bar{w}_2(\beta)+3}}. \tag{2.5}$$

Applying the framework and using the values of (2.4) then yields $f_1(\bar{w}_1(\beta), \bar{w}_2(\beta))$ in terms of $\bar{w}_2(\beta)$ only, i.e.,

$$f_1^*(\bar{w}_2) = \frac{1}{1 + e^{2\bar{w}_2-1.4}} + \frac{1}{1 + e^{-2\bar{w}_2+3}}. \tag{2.6}$$

This function is plotted in Figure 2.2b. In Figure 2.2a, $f_1(\bar{w}_1(\beta), \bar{w}_2(\beta))$ is plotted as a function of β; this figure results from simulation (see Section 1.5), as it is notoriously complex to find analytical results for $\bar{w}_2(\beta)$. We see from the figures that both graphs have equal characteristics. Both graphs, for instance, have the same number of minima (i.e., one). Also, the ranges of both graphs are the same. So, $f_1^*(\bar{w}_2)$ helps identifying the minimum value of the objective function and determining how much this value differs from the values in the endpoints. As opposed to $f_1(\bar{w}_1(\beta), \bar{w}_2(\beta))$, however, we can plot $f_1^*(\bar{w}_2)$ right away.

Nevertheless, we cannot conclude from the behavior of $f_1^*(\bar{w}_2)$ at what β-value this minimum occurs (say β_{min}). We only know that $\bar{w}_2(\beta_{\text{min}}) = 1.1$ at $F = 0.62$. So for instance if we are satisfied with the objective function within 2% of its minimum (i.e. F smaller than 0.6324), we then argue that we need $\bar{w}_2$ to be in the interval $[0.9, 1.3]$. This subsequently is the stopping criterion for a simulation or approximation procedure on the function $\bar{w}_2(\beta)$. As can be seen from Figure 2.2a, the procedure should result in a β_{opt} value in the interval $[0.71, 0.77]$, as to have a value for the objective function within 2% of the minimum.

From Figure 2.2b, we would assume to start the, in case of simulations, time-consuming quest for β_{min} at $\beta = 0$, as the minimum is more to the left of the graph. However, from Figure 2.2a, we observe that β_{min} is closer to 1. The information contained in Figure 2.2b can nonetheless be used to optimize the quest for β_{min} as it provides a means to assess which points of the curve you are probing. For instance, it is clear from Figure 2.2b that there will be only one local minimum (in terms of the objective function). Hence, we can use optimization procedures that are known to get stuck in local minima while searching the global minimum (see Section 2.2.4). In fact, these procedures, can be accelerated even further, as we know the actual minimum value of the objective function. Therefore, we know, while simulating, how far we are off and we can use this to our advantage. In Section 2.2.4, we describe and compare some procedures to find β_{min}.

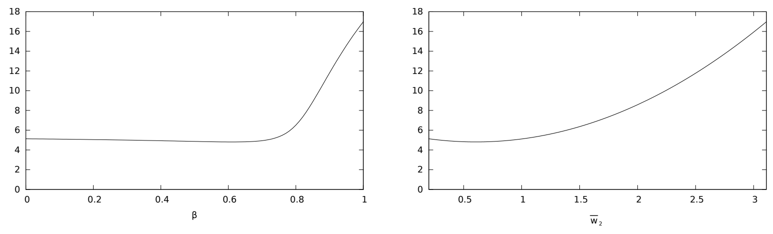
As a second example, we examine the objective function

$$f_2(\bar{w}_1(\beta), \bar{w}_2(\beta)) \triangleq (0.5\bar{w}_1(\beta))^2 + (1.3\bar{w}_2(\beta))^2. \tag{2.7}$$

Applying the framework and using the values in (2.4) then leads to

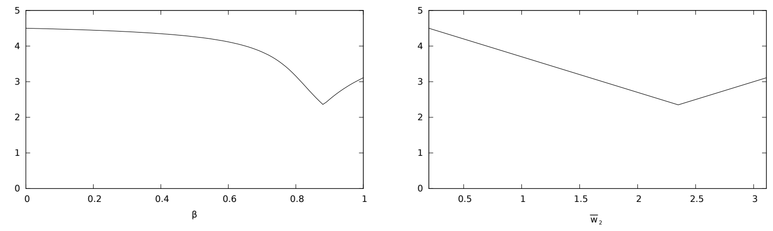
$$f_2^*(\bar{w}_2) = (2.35 - 0.5\bar{w}_2)^2 + (1.3\bar{w}_2)^2. \tag{2.8}$$

Plots for this second example are found in Figure 2.3. Function $f_2^*(\bar{w}_2)$ provides us with the value of the objective function in its minimum (see Figure 2.3b). The difference between the value of the objective function at $\beta = 0$ (5.13) and the minimum value (4.81)



(a) $f_2(\bar{w}_1(\beta), \bar{w}_2(\beta))$ (b) $f_2^*(\bar{w}_2)$

Figure 2.3: Comparison between the objective function $f_2(\bar{w}_1(\beta), \bar{w}_2(\beta))$ and $f_2^*(\bar{w}_2)$



(a) $f_3(\bar{w}_1(\beta), \bar{w}_2(\beta))$ (b) $f_3^*(\bar{w}_2)$

Figure 2.4: Comparison between the objective function $f_3(\bar{w}_1(\beta), \bar{w}_2(\beta))$ and $f_3^*(\bar{w}_2)$

is perhaps not worth the effort to search for β_{min} (the difference is 6.6%). Figure 2.3b can thus be used in advance to decide whether significant gain can be won by searching β_{min} compared to using the priority cases $\beta = 0$ or $\beta = 1$.

If, on the other hand, we make the same reasoning as in the previous example, thus allowing at most 2% deviation of the minimum, we need the objective function F to be smaller than 4.90 and consequently $\bar{w}_2$ in $[0.38, 0.83]$. An accompanying simulation/approximation method should then result in a β_{opt}-value in the interval $[0.44, 0.69]$.

The reasoning used here can be applied for an arbitrarily complex objective function in $\bar{w}_j(\beta)$. Indeed, every objective function minimization problem can be translated to a problem of finding the corresponding β for a certain performance vector (with or without some error margin). This effectively simplifies procedures; we will come back to this in Section 2.2.4.

Finally, we consider the objective function

$$f_3(\bar{w}_1(\beta), \bar{w}_2(\beta)) \triangleq \max(\bar{w}_1(\beta), \bar{w}_2(\beta)). \tag{2.9}$$

With the parameter values from (2.4) and the framework, this produces

$$f_3^*(\bar{w}_2) = \max(4.7 - \bar{w}_2, \bar{w}_2). \tag{2.10}$$

Figure 2.4 shows the plots of these functions. The left figure results from simulation; the right figure can readily be plotted for $\bar{w}_2 \in \Omega_2$. Figure 2.4 nicely shows the correspondence between plotting $f_3(\bar{w}_1(\beta), \bar{w}_2(\beta))$ as a function of $\bar{w}_2$ and plotting $f_3(\bar{w}_1(\beta), \bar{w}_2(\beta))$ as a function of β. Due to the increasing and decreasing character of $\bar{w}_2(\beta)$ and $\bar{w}_1(\beta)$, the max-function plots the first part of $\bar{w}_1(\beta)$ followed by the last part of $\bar{w}_2(\beta)$, respectively. The transition between the two parts is the point where $\bar{w}_1(\beta) = \bar{w}_2(\beta)$. This point matches the minimum of $f_3(\bar{w}_1(\beta), \bar{w}_2(\beta))$. Note that for other values of the arrival process parameters, either only $\bar{w}_1(\beta)$ or only $\bar{w}_2(\beta)$ may be depicted; this happens when, despite the increasing or decreasing character of both performance characteristics, one of the two is always greater than the other.

Note: Critical for this method is the availability of the values for $\bar{w}_j(0)$ and $\bar{w}_j(1)$ $(j = 1, 2)$. However, it does not matter how these values are obtained. For strict priority systems, many analytical results are available; this is, for instance, the case for the arrival and service processes we have used in the examples above (see [143]). For more complex arrival and/or service processes, this is not necessarily the case. To still use the framework, we can, for example, simulate the system for $\beta = 0$ and $\beta = 1$ as a start.

2.2.2 Convex Combination of Increasing Functions

Now, we take a step back and concentrate on the minimization of a specific class of objective functions. We derive some a priori results that are consequences of the GPS system dynamics and the choice of the objective function. To derive these results, the framework of the previous section is not required. In concreto, we study a convex combination of strictly increasing and sufficiently differentiable functions g_j of the mean unfinished work in both queues, i.e.,

$$F(\beta, \gamma) \triangleq \gamma g_1\big(\bar{w}_1(\beta)\big) + (1 - \gamma) g_2\big(\bar{w}_2(\beta)\big), \tag{2.11}$$

with $0 \le \gamma \le 1$. The parameter γ expresses the relative importance that is given to $\bar{w}_1(\beta)$: the higher γ, the more important $\bar{w}_1(\beta)$.

When $\gamma = 0$, the objective function only takes into account $\bar{w}_2(\beta)$; when $\gamma = 1$, only $\bar{w}_1(\beta)$ plays a role. In these cases, it is obvious that the strict priority scheduling discipline ($\beta = 0$ and $\beta = 1$, respectively) minimizes $F(\beta, \gamma)$, because of Theorem 2.1 and because of the assumption that g_1 and g_2 are increasing functions. Function g_j ($j = 1, 2$) expresses how increments in the mean unfinished work in queue j are penalized. For convex functions, for instance, increments of high values are more penalized than increments of low values. For concave functions, it is the other way around.

We first prove an important lemma.

Lemma 2.3. *Assume* $g_j'(x) > 0, \forall x \in \Omega_j$ *($g_j(x)$ is strictly increasing in the relevant domain). Then* $\frac{\partial F(\beta,\gamma)}{\partial \beta} > (<, =)\ 0$ *if and only if* $\gamma < (>, =)\ \phi(\beta)$ *with*

$$\phi(\beta) \triangleq \frac{g_2'(\bar{w}_2(\beta))}{g_2'(\bar{w}_2(\beta)) + g_1'(\bar{w}_1(\beta))}. \tag{2.12}$$

Furthermore, $\phi(\beta) \in]0, 1[$.

Proof. By taking the partial derivative of $F(\beta, \gamma)$ with respect to β and using Corollary 2.1, we obtain

$$\frac{\partial F(\beta,\gamma)}{\partial \beta} = \Big(g_2'(\bar{w}_2(\beta)) - \gamma [g_1'(\bar{w}_1(\beta)) + g_2'(\bar{w}_2(\beta))] \Big) \bar{w}_2'(\beta).$$

Then the lemma follows from the assumption that the functions g_j are strictly increasing functions and from Theorem 2.1 which states that $\bar{w}_2'(\beta) > 0$. □

Lemma 2.3 is important, as it separates, for given β, the interval $\gamma = [0, 1]$ in three parts, $[0, \phi(\beta)[$, the point $\phi(\beta)$, and $]\phi(\beta), 1]$, where the objective function $F(\beta, \gamma)$ increases, is constant and decreases, respectively. In particular, for $\beta = 0$, $F(\beta, \gamma)$ is increasing (decreasing) when $\gamma < (>)\ \phi(0)$; for $\beta = 1$, the objective function is increasing (decreasing) when $\gamma < (>)\ \phi(1)$. It is easily seen that $\phi(0)$ and $\phi(1)$ can be calculated from results of strict priority scheduling, see (2.12) (if these results are available). The values $\phi(0)$ and $\phi(1)$ are of primordial importance. Indeed, we prove that for certain important classes of functions g_j and depending on the value of γ with respect to $\phi(0)$ and $\phi(1)$, strict priority scheduling (either $\beta = 0$ or $\beta = 1$) is optimal.

Specifically, we analyze three particular classes for the (increasing) functions g_j namely linear functions, convex functions and concave functions. The derivative of $\phi(\beta)$ will play an important role in these analyses, so we first calculate this derivative from equation (2.12):

$$\phi'(\beta) = \frac{\Big(g_2''(\bar{w}_2(\beta))g_1'(\bar{w}_1(\beta)) + g_2'(\bar{w}_2(\beta))g_1''(\bar{w}_1(\beta))\Big)\bar{w}_2'(\beta)}{\Big(g_2'(\bar{w}_2(\beta)) + g_1'(\bar{w}_1(\beta))\Big)^2} \tag{2.13}$$

Theorem 2.2. *Assume $g_j'(x) > 0$ and $g_j''(x) = 0$, $\forall x \in \Omega_j$ (i.e., $g_j(x)$ is a linear increasing function). Then $\phi(\beta) = \phi_C$ is a constant function. As a result, strict priority is always optimal, namely*

(i) for $\gamma < \phi_C, \beta = 0$ (strict priority for class 2) is optimal,

(ii) for $\gamma > \phi_C, \beta = 1$ (strict priority for class 1) is optimal, and

(iii) for $\gamma = \phi_C$, all values of β are equally optimal.

Proof. From equation (2.13), it follows that $\phi(\beta)$ is a constant (say ϕ_C). Then the theorem follows directly from Lemma 2.3: (i) if $\gamma < \phi_C$, $F(\beta, \gamma)$ is strictly increasing with respect to β and has its minimum in $\beta = 0$; (ii) if $\gamma > \phi_C$, $F(\beta, \gamma)$ is strictly decreasing with respect to β and reaches its minimum in $\beta = 1$; and, (iii) if γ is equal to the constant ϕ_C, $F(\beta, \gamma)$ is identical for all values of β. $\square$

From this theorem, it follows that strict priority is always optimal when the objective function is a weighted combination of linear functions of the mean unfinished work in both queues. Similar observations have been made in the past (cf. the $c\mu$-rule, Section 2.3.2).

We now turn to convex increasing functions g_j.

Theorem 2.3. *Assume $g_j'(x) > 0$ and $g_j''(x) > 0$, $\forall x \in \Omega_j$ (i.e., $g_j(x)$ is strictly increasing and strictly convex). Then $\phi(\beta)$ is a strictly increasing function. As a result,*

(i) for $\gamma \leq \phi(0), \beta = 0$ (strict priority for class 2) is optimal,

(ii) for $\gamma \geq \phi(1), \beta = 1$ (strict priority for class 1) is optimal, and

(iii) for $\phi(0) < \gamma < \phi(1)$, $\beta_{opt}(\gamma) = \phi^{-1}(\gamma)$ is optimal, with ϕ^{-1} the inverse function of ϕ. The function $\beta_{opt}(\gamma)$ has the following properties: it is different from 0 or 1 (i.e., we have true GPS) and it is strictly increasing.

Proof. The right-hand side of equation (2.13) is positive, due to the assumptions on the functions g_j and Theorem 2.1. As a consequence, $\phi(\beta)$ is a strictly increasing function. It then follows directly that if

$\gamma \leq \phi(0)$, $\gamma < \phi(\beta)$ for $0 < \beta \leq 1$. From Lemma 2.3, we have that $F(\beta, \gamma)$ is in this case non-decreasing with respect to β and reaches its minimum in $\beta = 0$. If $\gamma \geq \phi(1)$, on the other hand, $\gamma > \phi(\beta)$ for $0 \leq \beta < 1$. In this case, $F(\beta, \gamma)$ is non-increasing with respect to β and reaches its minimum in $\beta = 1$. Finally, when $\phi(0) < \gamma < \phi(1)$, there is a unique $\beta \in]0, 1[$ so that $\phi(\beta) = \gamma$ (since $\phi(\beta)$ is a strictly increasing function). We denote this β by $\beta_{\text{opt}}(\gamma)$ (i.e., $\beta_{\text{opt}} = \phi^{-1}(\gamma)$). Then Lemma 2.3 indicates that $F(\beta, \gamma)$ decreases with β in the interval $[0, \beta_{\text{opt}}(\gamma)[$ and increases in the interval $]\beta_{\text{opt}}(\gamma), 1]$. As a result, $F(\beta, \gamma)$ is unimodal, and $\beta_{\text{opt}}(\gamma)$ is optimal and lies between 0 and 1 ($0 < \beta_{\text{opt}}(\gamma) < 1$). Finally, $\beta_{\text{opt}}(\gamma)$ being strictly increasing follows from $\beta_{\text{opt}}(\gamma) = \phi^{-1}(\gamma)$ and $\phi(\beta)$ strictly increasing. $\square$

From this theorem, it follows that GPS might be optimal for weighted combinations of convex increasing functions of the mean unfinished work in both queues. Moreover, the theorem sets bounds on the values of γ for which GPS is optimal. These bounds can be calculated from results of queueing analyses of a strict priority system. This is a big advantage, as the analysis of a strict priority system is usually much easier than the analysis of a GPS system. Outside the bounds, strict priority is optimal, so one does not have to search for the optimal β. Only inside the bounds, one should search for the optimal β. The optimal β is in fact $\phi^{-1}(\gamma)$, with $\phi(\beta)$ defined in (2.12). In order to calculate $\phi(\beta)$, however, we need to find the mean unfinished work in both queues for that β, which is the hard part. To find the optimal β, one can, for example, adopt the *golden section search* algorithm, as explained in subsection 2.2.4.

We can see an example of Theorem 2.3 in Figure 2.3. This is because we can reformulate the optimization problem (2.7) of f_2 as an optimization problem of a function of class $F(\beta, \gamma)$.

$$\begin{aligned} f_2(\bar{w}_1(\beta), \bar{w}_2(\beta)) &= (0.5\bar{w}_1(\beta))^2 + (1.3\bar{w}_2(\beta))^2 \\ &= \frac{1}{2}\left(\sqrt{2} \cdot 0.5 \cdot \bar{w}_1(\beta)\right)^2 + \frac{1}{2}\left(\sqrt{2} \cdot 1.3 \cdot \bar{w}_2(\beta)\right)^2 \end{aligned} \tag{2.14}$$

f_2 is thus of the form of $F(\beta, \gamma)$ with $\gamma = 0.5$ and the increasing convex functions $g_1(x) = (0.5\sqrt{2}x)^2$ and $g_2(x) = (1.3\sqrt{2}x)^2$. We thus calculate $\phi(0) = 0.10$ and $\phi(1) = 0.84$ using (2.12) and (2.4). We find that in this example $\gamma = 0.5$ is indeed in $]\phi(0), \phi(1)[$ and thus in case (iii) of Theorem 2.3 which states that there exists a single minimum for this objective function other than $\beta = 0$ or $\beta = 1$. This is indeed what we see in Figure 2.3.

Finally, we consider concave increasing functions g_j.

Theorem 2.4. *Assume $g_j'(x) > 0$ and $g_j''(x) < 0$, $\forall x \in \Omega_j$ (i.e., $g_j(x)$ is strictly increasing and strictly concave). Then $\phi(\beta)$ is a strictly decreasing function. As a result, strict priority is always optimal, namely*

(i) for $\gamma \leq \phi(1), \beta = 0$ (strict priority for class 2) is optimal,

(ii) for $\gamma \geq \phi(0), \beta = 1$ (strict priority for class 1) is optimal, and

(iii) for $\phi(1) < \gamma < \phi(0)$, $\beta = 0$ ($\beta = 1$) is optimal if $F(0,\gamma) <$ ($>$) $F(1,\gamma)$; $F(0,\gamma) = F(1,\gamma)$, both $\beta = 0$ and $\beta = 1$ are optimal.

Proof. From equation (2.13) and Theorem 2.1, it follows that $\phi(\beta)$ is strictly decreasing for strictly increasing and strictly concave functions g_j. By similar arguments as in the previous proof, we verify cases (i) and (ii). When $\phi(1) < \gamma < \phi(0)$, $F(\beta,\gamma)$ first increases with respect to β, until it reaches a maximum, and then decreases with respect to β. Consequently, the minimum of the objective function is reached for $\beta = 0$ or $\beta = 1$, depending on which is lowest. This results in case (iii). □

This theorem states that for weighted combinations of concave increasing functions of the mean unfinished work in both queues, strict priority scheduling is always optimal. One merely has to compare the objective functions for $\beta = 0$ and $\beta = 1$ (priority for class 2 and class 1, respectively).

Some Illustrative Examples

Here we apply and support the analytical derivations from the previous three theorems to some interesting case studies. Specifically, we study three interesting (classes of) objective functions $F(\beta,\gamma)$, corresponding to each of the three theorems. For these cases, we assume a general arrival process independent and identically distributed from slot to slot, as defined in Section 1.3. In the numerical examples shown in the figures, we chose the two-dimensional binomial arrival processes (see Section 1.6), additionally, we switched to packet delays instead of unfinished work. This demonstrates the conversion shown in the introduction of Section 2.2 and is practically most relevant in the context of heterogeneous services.

In [143] the authors derive expressions for the pgfs of the queue contents and the packet delays for a queueing model with two classes and class 1 having strict priority over class 2 (in our setting with

$\beta = 1$). This gives

$$\bar{w}_1(1) = \lambda_1 + \frac{\lambda_{11}}{2(1-\lambda_1)}, \tag{2.15}$$

and

$$\bar{w}_2(1) = \lambda_2 + \frac{\lambda_{TT}}{2(1-\lambda_T)} - \frac{\lambda_{11}}{2(1-\lambda_1)}, \tag{2.16}$$

for the mean unfinished work in both queues, where $\lambda_{jj} \triangleq \mathrm{E}[a_j^2]$ the second moment of the number of arrivals of class j, and $\lambda_{TT} \triangleq \mathrm{E}[a_T^2]$ the second moment of the total number of arrivals. To find the formulas for the same model but with class 2 having strict priority over class 1 (i.e., $\beta = 0$ in the GPS setting), one just has to replace $A(z_1, z_2)$ by $A(z_2, z_1)$ in the expressions for the pgfs and take the appropriate derivatives. Clearly the result is analog:

$$\bar{w}_1(0) = \lambda_1 + \frac{\lambda_{TT}}{2(1-\lambda_T)} - \frac{\lambda_{22}}{2(1-\lambda_2)}, \tag{2.17}$$

and

$$\bar{w}_2(0) = \lambda_2 + \frac{\lambda_{22}}{2(1-\lambda_2)}. \tag{2.18}$$

These are also the expressions we used to calculate (2.4). As described, these expressions are sufficient to calculate $\phi(0)$ and $\phi(1)$, which identify the values of γ for which either priority ($\beta = 0$ or $\beta = 1$) or GPS ($0 < \beta < 1$) is optimal.

Linear functions: We start with the case where the functions g_j are linear increasing functions. Assume $g_j(x) = a_j x$ ($j = 1, 2$), with a_j constants. Using equation (2.12) then yields

$$\phi(\beta) = \frac{a_2}{a_2 + a_1}. \tag{2.19}$$

As proved in Theorem 2.2, $\phi(\beta)$ is constant in this case. As a result, the objective function $F(\beta, \gamma)$ is strictly increasing w.r.t. β for $\gamma < \phi(\beta)$, constant for $\gamma = \phi(\beta)$, and decreasing for $\gamma > \phi(\beta)$. This is demonstrated in Figure 2.5, where we depict $F(\beta, \gamma)$ as a function of β, for $a_j = 1/\lambda_j$, $\alpha = 0.8$, $\lambda_T = 0.9$, and five different values of γ (including $\phi(\beta) = 0.8$). As already mentioned, $F(\beta, \gamma)$ cannot be calculated exactly for $0 < \beta < 1$. Estimates for $F(\beta, \gamma)$ (here, and further in this section), for 101 equidistant values of β in the

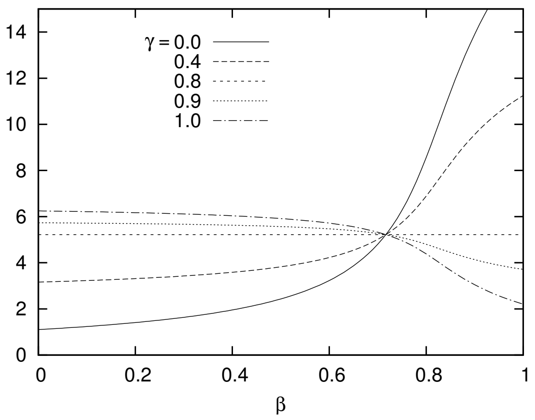


Figure 2.5: Objective functions $F(\beta, \gamma)$ as a function of β, for $g_j(x) = x/\lambda_j$ $(j = 1, 2)$, $\alpha = 0.8$, and $\lambda_T = 0.9$

range $[0, 1]$, are obtained through coupled Monte-Carlo simulations, as described in Section 1.5.

From Theorem 2.2, it follows that $\beta = 0$ is optimal if $\gamma < \phi(\beta)$, or, by means of formula (2.19), if

$$a_1\gamma < a_2(1 - \gamma). \tag{2.20}$$

This can be viewed as an occurrence of the $c\mu$-rule which states that the queue with the highest $c \cdot \mu$ should be given the highest priority [139], with c the weight given to the mean holding time of that queue in the objective function and μ the service rate of that queue. In our case, γ and $1 - \gamma$ play the role of c (the weights in the objective function) and $a_j = \mu_j$ if we are concerned with holding times (queue contents). For the special case $\mu_j = 1$, we find that class 2 should be given priority if $\gamma < 1/2$. We look in to the $c\mu$-rule further in Section 2.3.2. If we are interested in minimizing the weighted sum of the mean delays, class 2 should be given priority if $\gamma < \alpha$. This can also be observed from Figure 2.5. Note, finally, that all curves intersect at the same point. For the corresponding value of β, the objective function is independent of γ. This is only possible if $\bar{d}_1(\beta) = \bar{d}_2(\beta)$, i.e., if the mean delays for both classes are balanced.

Convex functions: Secondly, we demonstrate the case where the functions g_j are convex increasing functions. As mentioned earlier, the rationale for convex g_j is that increments of high values of the considered performance measures are penalized more than increments of low values. Assume $g_j(x) = (a_j x)^n$ $(j = 1, 2)$, with n discrete and larger than 1, and a_j constants. First, we treat the case $n = 2$; later, we consider general values of n.

$\underline{n = 2}$ By using equation (2.12) and expressions (2.15)-(2.18), we find that

$$\phi(0) = \frac{a_2^2(\lambda_{22} + 2\lambda_2(1-\lambda_2))(1-\lambda_T)}{\left\{ \begin{array}{l} (1-\lambda_T)(a_2^2 - a_1^2)\lambda_{22} + a_1^2(1-\lambda_2)\lambda_{TT} \\ +2(1-\lambda_T)(1-\lambda_2)(a_2^2\lambda_2 + a_1^2\lambda_1) \end{array} \right\}}, \tag{2.21}$$

and

$$\phi(1) = \frac{a_2^2 \left\{ \begin{array}{l} (1-\lambda_1)\lambda_{TT} - (1-\lambda_T)\lambda_{11} \\ +2\lambda_2(1-\lambda_T)(1-\lambda_1) \end{array} \right\}}{\left\{ \begin{array}{l} (1-\lambda_T)(a_1^2 - a_2^2)\lambda_{11} + a_2^2(1-\lambda_1)\lambda_{TT} \\ +2(1-\lambda_T)(1-\lambda_1)(a_2^2\lambda_2 + a_1^2\lambda_1) \end{array} \right\}}, \tag{2.22}$$

respectively. In this case, $\phi(0)$ and $\phi(1)$ are not equal. In fact, it follows from Theorem 2.3 that $\phi(0) < \phi(1)$. From (2.21)-(2.22), this can also be proved for this specific case. In Figure 2.6, we illustrate $\phi(0)$ and $\phi(1)$ as functions of α with $\lambda_T = 0.9$. For $\gamma \geq \phi(1)$ (i.e., at the upper left region of the graph), $\beta_{\text{opt}}(\gamma) = 1$ according to Theorem 2.3. When $\gamma \leq \phi(0)$ (i.e., at the lower right region of the graph), $\beta_{\text{opt}}(\gamma) = 0$. When $\phi(0) < \gamma < \phi(1)$, Theorem 2.3 states that the objective function $F(\beta, \gamma)$ reaches a minimum for some β between 0 and 1. So here, $\beta_{\text{opt}}(\gamma) \in]0, 1[$.

We illustrate these results by choosing a specific value for α and looking at the behavior of $F(\beta, \gamma)$ for that α. Assume $\alpha = 0.8$. Then $\phi(0) = 0.41$ and $\phi(1) = 0.96$. Figure 2.7 shows $F(\beta, \gamma)$ as a function of β, for $\alpha = 0.8$, $\lambda_T = 0.9$, and five different values of γ (including 0.41 and 0.96). We clearly see that only when $\gamma \in]\phi(0), \phi(1)[$, $F(\beta, \gamma)$ reaches a minimum for some β different from 0 and 1. For all other values of γ (i.e., $\gamma \in [0, \phi(0)]$ and $\gamma \in [\phi(1), 1]$), $F(\beta, \gamma)$ is non-decreasing and non-increasing respectively w.r.t. β and, hence, reaches a minimum in $\beta = 0$ and $\beta = 1$, respectively. For $\gamma = 0.41$ and $\gamma = 0.96$, it is observed that the objective functions have horizontal asymptotes in $\beta = 0$ and $\beta = 1$, respectively, as proved in Lemma 2.3. Furthermore, it can be seen from the curve for $\gamma = 0.9$ that the difference between GPS with $\beta = \beta_{\text{opt}}(\gamma)$ and the strict priority cases $\beta = 0$ and $\beta = 1$ can be considerable.

When $\gamma \in]\phi(0), \phi(1)[$, we need to search for the optimal value of β. Figure 2.8 illustrates the optimal values of β as a function of γ, for $\lambda_T = 0.9$, and three different values of α. The search for this

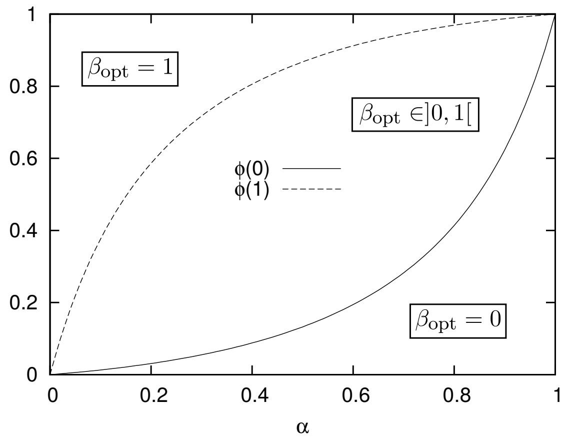


Figure 2.6: $\phi(0)$ and $\phi(1)$ as a function of α, for $a_j = 1/\lambda_j$ $(j = 1, 2)$ and $\lambda_T = 0.9$

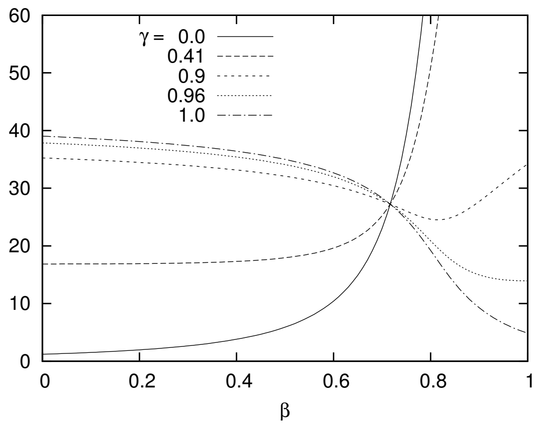


Figure 2.7: Objective functions $F(\beta, \gamma)$ as a function of β, for $g_j(x) = (x/\lambda_j)^2$ $(j = 1, 2)$, $\alpha = 0.8$, and $\lambda_T = 0.9$

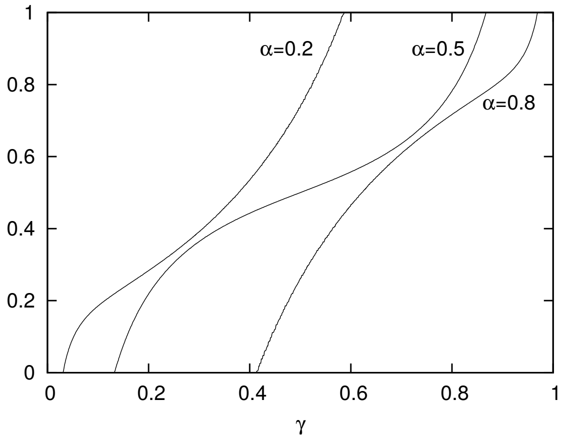


Figure 2.8: Optimal values of β as a function of γ, for $\lambda_T = 0.9$ and $g_j(x) = (x/\lambda_j)^2$ $(j = 1, 2)$

optimal value is thoroughly described in Section 2.2.4. Here, $\beta_{\text{opt}}(\gamma)$ was produced using golden section search on the objective function for 1001 equidistant values of γ and a β-precision of 0.0001. We observe that $\beta_{\text{opt}}(\gamma)$ increases the most for γ close to either $\phi(0)$ or $\phi(1)$, which shows that if $\gamma \in]\phi(0), \phi(1)[$, the optimal β is likely to be not that close to 0, nor to 1. This will be more pronounced for general powers n, which we briefly comment on next.

General n Figure 2.9 illustrates the optimal values of β as a function of γ, for $\alpha = 0.8$, $\lambda_T = 0.9$, and several values of n. The optimal values of β are obtained in the same way as in the previous figure (i.e., with the golden section search algorithm). We observe that the curves for $\beta_{\text{opt}}(\gamma)$ become steeper in the neighborhoods of $\phi(0)$ and $\phi(1)$ with increasing n[4]. This can be understood for $n \to \infty$, in particular. For $n \to \infty$, the value of γ becomes meaningless; the minimization of the objective function comes down to the minimization of the maximum of the mean packet delays for both classes, i.e., the balancing of the mean packet delays. In other words, for $n \to \infty$, $\beta_{\text{opt}}(\gamma)$ becomes independent of γ (except for $\gamma = 0$ and $\gamma = 1$ where $\beta = 0$ and $\beta = 1$ are still optimal) and this $\beta_{\text{opt}}(\gamma) \equiv \beta_{\text{opt}}$ makes $\bar{d}_1(\beta_{\text{opt}}) = \bar{d}_2(\beta_{\text{opt}})$. This effect is also observed in [118] in a different

[4] $\phi(0)$ can be read from the graph as the smallest value for γ where $\beta = 0$ is no longer optimal. For $n = 2$ this is 0.42 and for $n = 4$ it is 0.02. Vice versa, $\phi(1)$ is the smallest value for γ where $\beta = 1$ is optimal. For $n = 2$ this is 0.96 and for $n = 4$ it is 1.

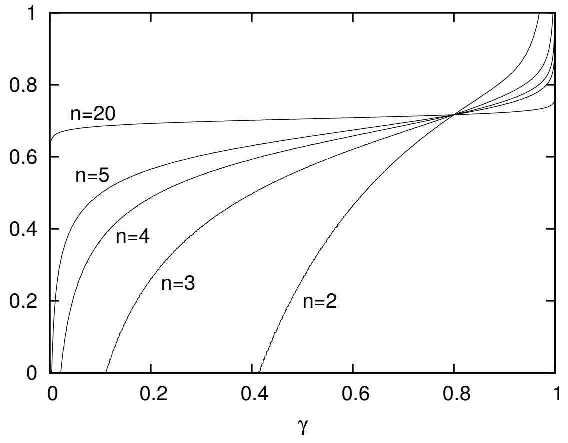


Figure 2.9: Optimal values of β as a function of γ, for $\alpha = 0.8$, $\lambda_T = 0.9$, and $g_j(x) = (x/\lambda_j)^n$ $(j = 1, 2)$

context. Note, furthermore, that β_{opt} is in fact the same β for which all curves in Figure 2.5 and Figure 2.7 intersect.

A second observation from Figure 2.9 is that $\phi(0)$ decreases and $\phi(1)$ increases with n. This is also clear from Figure 2.10, where we show the difference between $\phi(1)$ and $\phi(0)$ as a function of n, for $\lambda_T = 0.9$ and three different values of α (the curves for 0.2 and 0.8 are identical, due to the apparent symmetry). The difference indeed increases (rapidly) with n. For $n = 1$, the difference is 0, cf. the linear example; for $n \to \infty$, the difference tends to 1. So for larger n, GPS is more frequently the optimal scheduling discipline. For $\gamma = 0.2$ and $n = 2$, for example, strict priority (with $\beta = 0$) is optimal; for $\gamma = 0.2$ and $n = 3$, GPS is optimal. This basically means that, to determine the optimal β, the need for simulation increases when n increases. To counterbalance this, the optimal β becomes less sensitive to γ, if γ is not too close to either 0 or 1 (see Figure 2.9).

Concave functions: We end this section with the case where the functions g_j are concave increasing functions. In contrast with convex functions g_j, increments of low values of the considered performance measures are here more penalized than increments of high values. Assume $g_j(x) = \ln(a_j x)$ $(j = 1, 2)$, with a_j a constant. In the same way as above, we can easily obtain formulas for $\phi(0)$ and $\phi(1)$. According to Theorem 2.4, $\phi(0) > \phi(1)$. This is demonstrated in Figure 2.11, where we depict $\phi(0)$ and $\phi(1)$ as a function of α, for

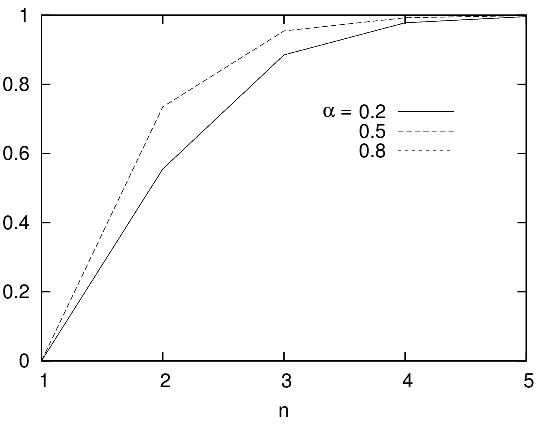


Figure 2.10: $\phi(1) - \phi(0)$ as a function of n, for $\lambda_T = 0.9$ and $g_j(x) = (x/\lambda_j)^n$ $(j = 1, 2)$

$\lambda_T = 0.9$. When $\gamma \geq \phi(0)$, $\beta_{\text{opt}}(\gamma) = 1$; when $\gamma \leq \phi(1)$, $\beta_{\text{opt}}(\gamma) = 0$. When $\phi(1) < \gamma < \phi(0)$, it depends on the values of $F(0, \gamma)$ and $F(1, \gamma)$ whether $\beta = 0$ or $\beta = 1$ is optimal. By using equation (2.11) and expressions (2.15)-(2.18), we can determine the conditions on γ so that $F(0, \gamma) < (>, =) F(1, \gamma)$. Indeed, solving the equation $F(0, \gamma) = F(1, \gamma)$ for γ leads to a formula for the γ for which both $\beta = 0$ and $\beta = 1$ are optimal. Let us denote this value of γ by γ^*. For the cost function studied here we find

$$\gamma^* = \frac{\ln\left(\frac{\bar{w}_2(0)}{\bar{w}_2(1)}\right)}{\ln\left(\frac{\bar{w}_2(0)\bar{w}_1(1)}{\bar{w}_1(0)\bar{w}_2(1)}\right)},$$

whereby we have omitted using (2.15) - (2.18) for space saving reasons and because the replacement does not present any significant simplifications. We illustrate the behavior of γ^* in Figure 2.11. When $\gamma \geq \gamma^*$, $\beta_{\text{opt}}(\gamma) = 1$. When $\gamma \leq \gamma^*$, on the other hand, $\beta_{\text{opt}}(\gamma) = 0$.

Finally, Figure 2.12 shows $F(\beta, \gamma)$ as a function of β, for $\alpha = 0.8$, $\lambda_T = 0.9$, and six different values of γ. Amongst those values of γ are $\phi(0)$ $(= 0.95)$ and $\phi(1)$ $(= 0.33)$. For these system and objective function parameters, furthermore, $\gamma^* = 0.72$. We clearly see that when $\gamma < 0.72$, $F(\beta, \gamma)$ reaches its minimum in $\beta = 0$. For $\gamma > 0.72$, $\beta = 1$ minimizes $F(\beta, \gamma)$. Furthermore, we observe that the curves corresponding with $\gamma \in]\phi(1), \phi(0)[$ reach a maximum for some $\beta \in]0, 1[$, as reasoned in the proof of Theorem 2.4.

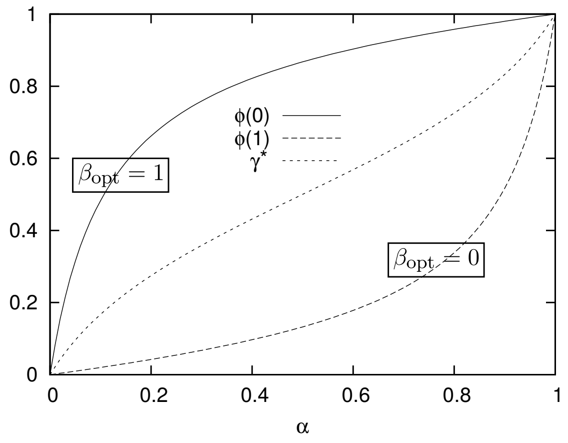


Figure 2.11: $\phi(0)$, $\phi(1)$, and γ^* as a function of α, for $g_j(x) = \ln(x/\lambda_j)$ and $\lambda_T = 0.9$

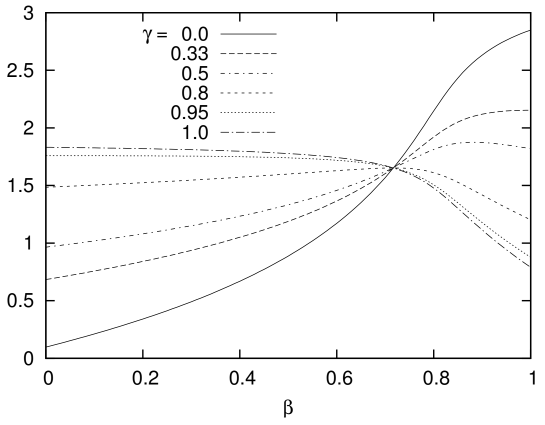


Figure 2.12: Objective functions $F(\beta, \gamma)$ as a function of β, for $\alpha = 0.8$, $\lambda_T = 0.9$, and $g_j(x) = \ln(x/\lambda_j)$ $(j = 1, 2)$

2.2.3 Framework for Convex Combination of General Functions

Now we generalize the results from the previous subsection using the framework from Section 2.2.1. We drop the requirement on g_j and only assume g_j to be *sufficiently* differentiable. This widens the class of objective functions of the form $F(\beta, \gamma)$. The goal is now to determine the behavior of this class of functions and to do a sensitivity analysis on γ.

We start by generalizing and promoting Lemma 2.3 to the following theorem.

Theorem 2.5. *Assume g_j is continuously differentiable in Ω_j. Then $\gamma = \phi(\beta)$ if and only if (iff) $\frac{\partial F}{\partial \beta}(\beta, \gamma) = 0$, with*

$$\phi(\beta) \triangleq \frac{g_2'(\bar{w}_2(\beta))}{g_2'(\bar{w}_2(\beta)) + g_1'(\bar{w}_1(\beta))}.$$

Proof. We find that $\frac{\partial F}{\partial \beta}(\beta, \gamma) = 0$ is equivalent with

$$[g_2'(\bar{w}_2(\beta)) - \gamma(g_1'(\bar{w}_1(\beta)) + g_2'(\bar{w}_2(\beta)))]\bar{w}_2'(\beta) = 0, \tag{2.23}$$

where we have used that $\bar{w}_1'(\beta) = -\bar{w}_2'(\beta)$, see Corollary 2.1. According to Theorem 2.1 ($\bar{w}_2'(\beta) > 0$) and using (2.23), this leads to

$$\gamma = \phi(\beta).$$

$\square$

The β-values for which $\phi(\beta) = \gamma$ are the critical points of the objective function $F(\beta, \gamma)$. Critical points can be either extrema or inflection points with a horizontal asymptote.

For simplicity, we assume $\phi(\beta)$ to be continuous. Discontinuities only occur for β-values for which $g_2'(\bar{w}_2(\beta)) = -g_1'(\bar{w}_1(\beta))$. For these β-values,

$$\frac{\partial F}{\partial \beta}(\beta, \gamma) = g_2'(\bar{w}_2(\beta))\bar{w}_2'(\beta). \tag{2.24}$$

As $\bar{w}_2'(\beta)$ is positive, the objective function will increase or decrease like g_2. We will disregard these cases in the remainder, as the discontinuities in $\phi(\beta)$ do not lead to special cases for $F(\beta, \gamma)$. The extensions are straightforward but only result in more involved expressions.

To be able to distinguish extrema from inflection points, we need higher-order derivatives of the objective function. These will allow us to perform the higher-order derivative test on the objective function. Therefore, we extend Theorem 2.5.

Theorem 2.6. *Assume g_j is n times continuously differentiable in Ω_j. Then $\phi^{(i-1)}(\beta) = 0, \forall i = 2, ..., n$, and $\phi(\beta) = \gamma$ iff $\frac{\partial^i F}{\partial \beta^i}(\beta, \phi(\beta)) = 0, \forall i = 1, ..., n$. Here, $\phi^{(j)}(\beta)$ denotes the j-th derivative of $\phi(\beta)$.*

Proof. (by induction) The base case $n = 1$ follows from Theorem 2.5. For the induction hypothesis, assume that $\phi(\beta) = \gamma$ and $\phi^{(i-1)}(\beta) = 0$ for $i = 2, ..., n-1$ iff $\frac{\partial^i F}{\partial \beta^i}(\beta, \phi(\beta)) = 0$ for $i = 1, ..., n-1$. To complete the theorem, we prove that $\phi^{(n-1)}(\beta) = 0$ iff $\frac{\partial^n F}{\partial \beta^n}(\beta, \phi(\beta)) = 0$. We find that

$$\begin{aligned}\frac{\partial^n F}{\partial \beta^n} =& \frac{\partial^{n-1}}{\partial \beta^{n-1}} \left(\frac{\partial F}{\partial \beta} \right) \\ =& \frac{\partial^{n-1}}{\partial \beta^{n-1}} \Big(\bar{w}_2'(\beta) \big(g_1'(\bar{w}_1(\beta)) + g_2'(\bar{w}_2(\beta)) \big) \big(\phi(\beta) - \gamma \big) \Big),\end{aligned}$$

where we have used (2.23). Define, furthermore, $\Delta(\beta, \gamma)$ as $\phi(\beta) - \gamma$ and $\chi(\beta)$ as $\bar{w}_2'(\beta)\big(g_1'(\bar{w}_1(\beta)) + g_2'(\bar{w}_2(\beta))\big)$. Then we write

$$\begin{aligned}\frac{\partial^n F}{\partial \beta^n} &= \frac{\partial^{n-1}}{\partial \beta^{n-1}} \Big(\chi(\beta) \Delta(\beta, \gamma) \Big) \\ &= \sum_{l=0}^{n-1} \binom{n-1}{l} \chi^{(n-1-l)}(\beta) \frac{\partial^l \Delta}{\partial \beta^l}(\beta, \gamma).\end{aligned}$$

Now, iff $\frac{\partial^l \Delta}{\partial \beta^l}(\beta, \gamma) = \frac{\partial^l (\phi(\beta) - \gamma)}{\partial \beta^l} = \frac{\partial^l \phi(\beta)}{\partial \beta^l} = \phi^{(l)}(\beta) = 0$ for $l = 1, ..., n-2$, and $\Delta(\beta, \gamma) = 0$ (i.e. the induction hypothesis),

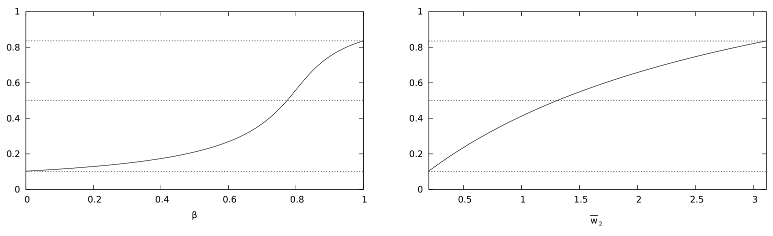
$$\begin{aligned}\frac{\partial^n F}{\partial \beta^n} &= \binom{n-1}{n-1} \chi^{(0)}(\beta) \frac{\partial^{n-1} \Delta}{\partial^{n-1} \beta}(\beta, \gamma) \\ &= \bar{w}_2'(\beta) \big(g_1'(\bar{w}_1(\beta)) + g_2'(\bar{w}_2(\beta)) \big) \phi^{(n-1)}(\beta).\end{aligned}$$

Strict monotonicity of $\bar{w}_2(\beta)$ and the continuity assumption of $\phi(\beta)$ that we made earlier prove that $\phi^{(n-1)}(\beta) = 0$ iff $\frac{\partial^n F}{\partial \beta^n}(\beta, \phi(\beta)) = 0$. This together with the induction hypothesis proves the induction step and thus concludes the proof of the theorem. □

The next corollary follows directly from Theorem 2.6:

Corollary 2.3. *If $\gamma = \phi(\beta)$, $\phi^{(1)}(\beta) = \cdots = \phi^{(n)}(\beta) = 0$ and $\phi^{(n+1)}(\beta) \neq 0$, then $F(\beta, \gamma)$ has a local extremum at β if n is even and an inflection point at β if n is odd.*

With this corollary, we can determine the behavior of $F(\beta, \gamma)$ by studying the behavior of $\phi(\beta)$. Suppose, for instance, that $\phi(\beta)$



(a) $\phi(\beta)$ (b) $\phi^*(\bar{w}_2)$

Figure 2.13: For F_4: Comparison between $\phi(\beta)$ and $\phi^*(\bar{w}_2)$

has one inflection point $\hat{\beta}$ with horizontal tangent. Then $\phi^{(2)}(\hat{\beta}) = \phi^{(1)}(\hat{\beta}) = 0$ and $\phi^{(3)}(\hat{\beta}) \neq 0$, and, as a consequence, $F(\beta, \gamma)$ has an extremum at $\hat{\beta}$ when $\gamma = \phi(\hat{\beta})$.

Unfortunately, we do not have a formula for $\phi(\beta)$, as we do not have explicit analytical results for the functions $\bar{w}_j(\beta)$. This is where the framework of Section 2.2.1 comes into play. In particular, the function $\phi(\beta)$ can be translated into a function in terms of $\bar{w}_2(\beta)$ instead of β (i.e. $\phi^*(\bar{w}_2)$ in the remainder). As there is a one-to-one mapping between the values in Ω_2 and the values in $[0, 1]$, we find that

$$\phi^*(\bar{w}_2) = \frac{g_2'(\bar{w}_2)}{g_2'(\bar{w}_2) + g_1'(\bar{w}_T - \bar{w}_2)}.$$

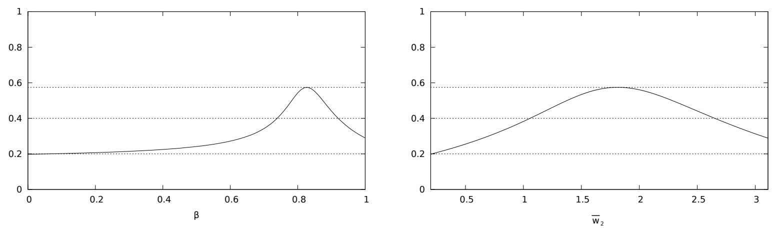
Using the framework, the previous corollary is reformulated as follows:

Corollary 2.4. *If $\gamma = \phi^*(\hat{\bar{w}}_2)$, ${\phi^*}^{(1)}(\hat{\bar{w}}_2) = \cdots = {\phi^*}^{(n)}(\hat{\bar{w}}_2) = 0$ and ${\phi^*}^{(n+1)}(\hat{\bar{w}}_2) \neq 0$, then $F^*(\hat{\bar{w}}_2, \gamma)$ has a local extremum at $\hat{\bar{w}}_2$ if n is even and an inflection point at $\hat{\bar{w}}_2$ if n is odd. As $\bar{w}_2(\beta)$ is bijective on $[0, 1]$ (Corollary 2.2) , $F(\beta, \gamma)$ also has a local extremum at $\hat{\beta}$ (with $\bar{w}_2(\hat{\beta}) = \hat{\bar{w}}_2$) if n is even and an inflection point at $\hat{\beta}$ if n is odd.*

Hereby, we defined $F^*(\bar{w}_2, \gamma)$ analogously to the other functions marked with a star, using the framework presented earlier. Now, we look at some examples to demonstrate the use of this corollary.

As a first example, we have composed figures to compare $\phi(\beta)$ (see Figure 2.13a) with $\phi^*(\bar{w}_2)$ (see Figure 2.13b). For these figures, we have used the objective function

$$F_4(\beta, \gamma) = \gamma(0.5\sqrt{2}\bar{w}_1(\beta))^2 + (1 - \gamma)(1.3\sqrt{2}\bar{w}_2(\beta))^2. \qquad (2.25)$$



(a) $\phi(\beta)$ (b) $\phi^*(\bar{w}_2)$

Figure 2.14: For F_5: Comparison between $\phi(\beta)$ and $\phi^*(\bar{w}_2)$

and the same two-dimensional binomial arrival process as in Section 2.2.1. It should be noticed that $F_4(\beta, 0.5) = f_2(\bar{w}_1(\beta), \bar{w}_2(\beta))$, see (2.14). We can draw similar conclusions as with the comparisons in Section 2.2.1. For instance, we can see that both graphs have the same range.

Using the aforementioned corollary, we see that for $\gamma \in [0.1, 0.83]$, $F_4(\beta,\gamma)$ reaches an extremum at a β-value different from 0 or 1. In particular for $\gamma \in [0.1, 0.83]$, there is a β-value and corresponding $\bar{w}_2(\beta)$-value, say $\hat{\beta}$ and $\bar{w}_2(\hat{\beta})$, respectively, for which $\gamma = \phi(\beta) = \phi^*(\bar{w}_2(\beta))$. Visually, this $\hat{\beta}$ and $\bar{w}_2(\hat{\beta})$ can be presented in the Cartesian coordinate systems $(\beta, \phi(\beta))$ and $(\bar{w}_2, \phi^*(\bar{w}_2))$, by drawing a horizontal line at the chosen value of γ (see Figure 2.13b); the intersection points of the horizontal lines and curves of $\phi(\beta)$ and $\phi^*(\bar{w}_2)$ then yield $\hat{\beta}$ and $\bar{w}_2(\hat{\beta})$, respectively.

Now according to Theorem 2.6, $\frac{\partial F_4}{\partial \beta}(\hat{\beta},\gamma) = 0$ and we have an extremum at $\hat{\beta}$ if $\frac{\partial^2 F_4}{\partial \beta^2}(\hat{\beta},\gamma) \neq 0$ or, equivalently, if $\phi^{(1)}(\hat{\beta}) \neq 0$. From Figure 2.3, where we have depicted $F_4(\beta,\gamma)$ for $\gamma = 0.5$ (i.e., $f_2(\bar{w}_1(\beta, \bar{w}_2(\beta)))$, we can see that $F_4(\beta,\gamma)$ is decreasing at $\beta = 0$ for $\gamma \in [0.1, 0.83]$. As we have a $\hat{\beta}$ for which $\gamma = \phi(\beta)$ in that interval, $F_4(\beta,\gamma)$ has an extremum, which is necessarily a minimum. Summarized, the couples $(\beta, \phi(\beta))$, indicated by the curve in the figure, are parameter combinations for (β,γ) that minimize the objective function. For $\gamma < 0.1$, there is no β for which $\gamma = \phi(\beta)$ and thus, according to Theorem 2.5, $F_4(\beta,\gamma)$ has no extremum between 0 and 1. Since $F_4(\beta,\gamma)$ is increasing at $\beta = 0$, $F_4(\beta,\gamma)$ is increasing with respect to all β. Analogously, for $\gamma > 0.83$, the objective function is decreasing with respect to β.

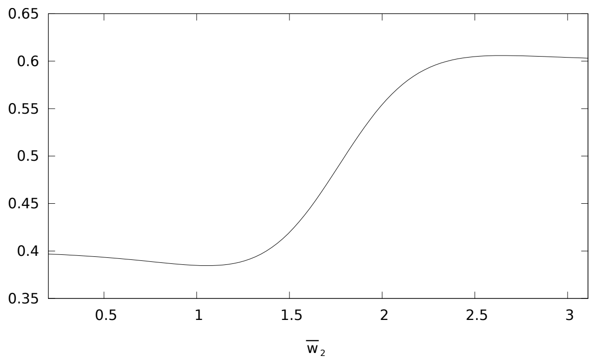


Figure 2.15: Objective function $F_5^*(\bar{w}_2, \gamma)$ for $\gamma = 0.4$

As a last example, we look at the objective function

$$F_5(\beta, \gamma) = \frac{\gamma}{1 + e^{-3\bar{w}_1(\beta)+9}} + \frac{1-\gamma}{1 + e^{-4\bar{w}_2(\beta)+7}}. \tag{2.26}$$

Using the same arrival processes and the same arrival process parameters as before, we depict the corresponding $\phi(\beta)$ in Figure 2.14a and $\phi^*(\bar{w}_2)$ in Figure 2.14b. Recall that the former is obtained via simulations, while the latter can be drawn directly. We can make the same reasoning as before. $\phi(\beta)$ and $\phi^*(\bar{w}_2)$ will have intersection points with horizontal lines at $\gamma \in [0.2, 0.58]$ only. For a γ-value in this interval, Theorem 2.5 dictates that $F_5(\beta, \gamma)$ will have extrema or inflection points. Using the corollary of Theorem 2.6, we know that inflection points only occur when also $\phi'(\beta) = 0$, so when $\phi(\beta)$ and $\phi^*(\bar{w}_2)$ have an extremum. For the example at hand, this occurs for $\gamma = 0.58$ and $\bar{w}_2 = 1.8$ (and from simulation, $\beta = 0.82$).

At $\bar{w}_2 = 0.2$ ($\beta = 0$), $F_5^*(\bar{w}_2, \gamma)$ is decreasing if $\gamma > 0.2$ and increasing if $\gamma < 0.2$ (this can easily be seen from (2.26)). So if we take $\gamma = 0.4$, we find that $F_5^*(\bar{w}_2, \gamma)$ is decreasing at $\bar{w}_2 = 0.2$. Furthermore, $F_5^*(\bar{w}_2, 0.4)$ reaches an extremum at $\bar{w}_2 = 1.1$ because at that $\bar{w}_2$-value $\phi^*(\bar{w}_2)$ intersects with a horizontal line at 0.4 (see Figure 2.14b). This extremum is necessarily a minimum. For higher values of $\bar{w}_2$, $F_5^*(\bar{w}_2, \gamma)$ is increasing again. At $\bar{w}_2 = 2.7$, we once more have a point of intersection between $\phi^*(\bar{w}_2)$ and the horizontal line at 0.4, and, hence, $F_5^*(\bar{w}_2, \gamma)$ has a second extremum, in this case a maximum. These conclusions can be verified in Figure 2.15, where we have plotted $F_5^*(\bar{w}_2, 0.4)$.

Using similar arguments for other values of γ, we constructed an annotated version of Figure 2.14a in Figure 2.16. In this figure, the

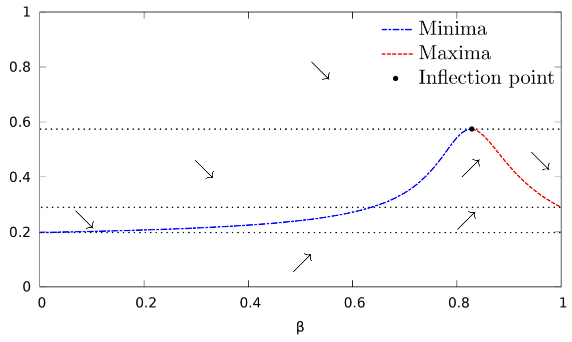


Figure 2.16: Annotated version of Figure 2.14a

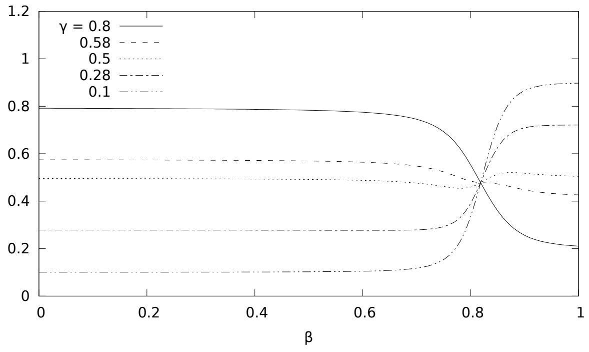


Figure 2.17: Simulation of F_5 for several γ

arrows indicate the behavior of $F_5(\beta, \gamma)$ for the (β, γ)-values in that area. We see that $F_5(\beta, \gamma)$ is decreasing above the curve of $\phi(\beta)$ and increasing under the curve. In fact, if we draw a path in the unit square (a collection of couples for (β, γ)), we know that the behavior of the objective function is indicated by the arrows in the figure. Furthermore, the sign (and thus the behavior) of $\frac{\partial F_5}{\partial \beta}(\beta, \gamma)$ will only change if the path intersects with $\phi(\beta)$. At the intersection point, the sign will change and the objective function will have an extremum or an inflection point. We can thus conclude that from a plot of $\phi^*(\bar{w}_2)$, we can determine the complete behavior of the objective function without running any simulation or relying on possibly inaccurate approximate expressions for the performance characteristics.

In Figure 2.17, finally, we have graphs of $F_5(\beta, \gamma)$, for several γ; they are obtained through simulation. We have chosen a γ-value from each area in Figure 2.16 and we see that the graphs in Figure 2.17 confirm our analysis. For $\gamma = 0.5$, for instance, we see that $F_5(\beta, \gamma)$ first reaches a minimum and later a maximum; for $\gamma = 0.58$, $F_5(\beta, \gamma)$ has an inflection point. We see from this study that the behavior of the objective function largely depends on the coefficient γ in that function. Using our theorems presented here, these different behaviors can be seen at a glance from a graph of $\phi^*(\bar{w}_2)$.

Let us now return to the sensitivity analysis. From Figure 2.14b, an analyst can see what the impact of a variation in γ will be on $\bar{w}_2$ (and subsequently also on the value of the objective function). An annotated version as presented in Figure 2.16 easily shows how the objective function behaves for different γ. This figure can be used to make a selection of different values of γ for which the objective function can be studied more closely, as was done in Figure 2.17. The mapping from $\bar{w}_2$ to β, however, is still unknown. To get this information, one needs to resort to either simulation or approximations. Part II of this dissertation is devoted to approximation analysis and in the next section, we show how simulations can be done efficiently.

2.2.4 Achieving the optimum

In this section, we combine the results obtained in the previous sections to optimize the simulation and optimization procedure. We will optimize objective functions of the form of (2.11) without the restriction on g_j being an increasing function (like in the previous section). We do this by optimizing for 101 values of γ equally spaced in $[0, 1]$. We compare several techniques and see how they influence the simulation effort and execution time. Lastly, we give some closing remarks on how the procedure could be sped up even further. If you only need to optimize a simple objective function (without changing the γ parameter) the optimization is even faster, though completely analog to the one presented here.

For the numerical results in this section, we optimized the objective function F_4. During the simulation runs for different γ, we only do one simulation for a given value of β. A table in memory holds the already simulated values for β and their results. This is a first way to speed up the process.

The search for the optimum (given that there is one, see previous subsections) boils down to the determination of the minimum of a function which cannot be computed exactly and thus has to be estimated. A first, albeit naive, method is to just simulate equally spaced β's in $[0, 1]$, with the spacing tailored to the required precision. We call this the *brute force method.*

If the function is strictly unimodal there are some well-known techniques for this optimization problem. In particular 'golden section search', 'successive parabolic interpolation', or a combination of these two, and 'stochastic approximation' (e.g., Robbins-Monro and Kiefer-Wolfowitz), have proven their merit in the past (see, e.g., [31, 73, 94] for more information). As a second method, we have used the 'golden section search' algorithm on the objective function[5]. This algorithm is known to be reliable [95] and easy to implement. Convergence to the global minimum is relatively slow but certain.

Golden section search & binary search

The golden section search algorithm [114] on a unimodal function $h(x)$ (that can not be evaluated but needs simulations) can be summarized as follows. In each iteration, we start with a triplet (x_1, x_2, x_3) of values of x ($x_1 < x_2 < x_3$) with $h(x_2) < h(x_1)$ and $h(x_2) < h(x_3)$. So the minimum of $h(x)$ lies inside the interval $[x_1, x_3]$ and the current estimation of this minimum is x_2. The next x (say x_4) is chosen in the largest of the two intervals $[x_1, x_2]$ and $[x_2, x_3]$, at a distance of $0.38197 = 1 - \frac{1}{\phi}$ (with ϕ the golden ratio) times this interval from x_2. x_4 splits the largest of these intervals, creating two intervals that relate to each other according to the golden ratio. Then, $h(x)$ is estimated for $x = x_4$ via Monte-Carlo simulation. Depending on the value of $h(x_4)$ relative to that of $h(x_2)$, the triplet (x_1, x_2, x_3) is updated (x_4 becomes an element of this triplet), leading to a narrower search interval. As termination condition for this algorithm, we choose to test the gaps between the values of x_1 and x_3. The algorithm terminates when the relative accuracy bounds $|x_3 - x_1| < \tau$, where we define τ as the x-precision. It is important to remark that estimates of $h(x)$ are still obtained through coupled Monte-Carlo simulations for different values of x.

Essentially the algorithm maintains an interval $[x_1, x_2]$ in which it knows the minimum is located. Two intermediate points are selected x_3 and x_4 after which it can shorten the interval to either $[x_1, x_4]$ or $[x_3, x_2]$ (given that $x_3 < x_4$) as we can identify the interval in which the minimum lies by evaluating the unknown function for x_3 and x_4. Instead of using the golden ratio to determine x_3 and x_4 in the interval, we can just uniformly distribute them, i.e., at one and two thirds of the length of the interval. In that case the algorithm is known as 'ternary search'.

Say we have a function $h(x)$, strictly monotone in the domain of interest, that can not be evaluated but needs simulation, and we have to determine for which value of x it achieves a certain value. In that case we can use a simpler version of the golden section search algorithm. In the interval $[x_1, x_2]$ we select x_3 according to the golden ratio. We simulate $h(x_3)$ and compare to $h(x_1)$ and $h(x_2)$. As h is monotone we can select in which of the two intervals $[x_1, x_3]$ or $[x_3, x_2]$ the value of interest is located.

Again instead of using the golden ratio the interval can simply be divided in two equal parts. In that case the interval is known as binary search.

All algorithms explained in this sidenote are standard algorithms for numerical analysis. They are examples of what is known in computer science as divide and conquer algorithms.

[5]This method was used for the optimization in Section 2.2.2, for instance, in Figures 2.8 and 2.9.

Table 2.1: Simulation times

Method	**Stopping criterium**	**# simulations**
Brute force	β-precision: 0.0001	10001
Golden section on F	β-precision: 0.0001	897
Golden section on $\bar{w}_2$	β-precision: 0.0001	617
Golden section on $\bar{w}_2$	β-precision: 0.0001	12
	or F-precision: 1%	

This method, however, is only usable for objective functions known to have a single extremum. This is, for instance, the case for F_4 as it has convex g_j (see Theorem 2.3). A third method, that is usable for all objective functions, follows from the general framework presented in Section 2.2.1. We know that it is not needed to work directly with the objective function. From the objective function, the optimal value of $\bar{w}_2$ can be calculated. Subsequently, the corresponding value of β to achieve this value of $\bar{w}_2$ needs to be obtained by simulation. Knowing that $\bar{w}_2(\beta)$ is monotonically increasing, we can use a simplified version of the golden section search method (see note).

We can use two different stopping criteria. The first one is the β-precision, as described in the note on golden section search. This is the size of the interval $x_2 - x_1$; if this value is small enough, we stop the algorithm and use the average $(x_1 + x_2)/2$ as value for β_{opt}. Another option, called the F-precision, is to stop the simulation once we found a β that leads to a *reasonable* value of F. This F-precision is the percentage deviation from the minimum of the objective function F (which we can calculate in advance, as shown in Section 2.2.1).

The computational effort of the different simulation methods for this specific case can be found in Table 2.1. Cases can be engineered where the efficiency order of these methods is different; however, these cases are exceptional and need to be tailor-made. Furthermore, golden section search on F has a limited usability as it can only be used for objective functions with a single extremum, specifically a minimum. It is clear that in general the more information and knowledge you have about the queueing system and objective function, the less simulations are needed. This often leads to complex algorithms that are only usable in a limited number of cases. The framework presented here, being as simple and general as it is, leads to large simulation gains without significant increase in complexity.

The methods presented here can be improved even further. Instead of using the golden section search in its purest form, the table with the already simulated β's could be used to select a starting interval $[x_1, x_2]$ after which golden section search could be used to further refine the result. This would lead to faster convergence. Another method (variation on golden section search), is to use multiple cores and select multiple x_3's. This way the interval $[x_1, x_2]$ will shrink much faster. Lastly, one could also vary the number of simulated slots as we get further into the algorithm, simulating less slots (and having a rougher estimation) when the interval $[x_1, x_2]$ is still large. A word of caution however is in order here, as this also induces extra variance. This variance could cause the algorithm to exclude the minimum. Using the presented insights the algorithm can easily detect when this happens and act accordingly.

2.3 Extension to general work-conserving parameterized schedulers

The framework we presented in Section 2.2.2 has a wider application range than only GPS as presented in the previous section. In this section, we show that the application of the framework can be extended to other examples of which we detail two specific examples namely the $c\mu$-rule we mentioned before and a semi-preemptive priority queue. This is a small side-step from the main topic of this dissertation, i.e. the study of discrete-time GPS, nevertheless it provides more insight into the mechanism of the framework.

2.3.1 Framework

We notice that the proofs of the most important theorems (Lemma 2.3 and Theorems 2.2-2.4) of the framework in the previous section are based on just two properties of the performance characteristic $\bar{w}_j$, i.e. Corollary 2.1 and Theorem 2.1. As a result, in whatever setting these properties are valid, the theorems from the framework are applicable. We now formalize this setting.

Consider a queueing system where tasks of two user classes, named 1 and 2, have to be executed. We adopt a work-conserving scheduling policy with parameter $\beta \in [0, 1]$ expressing the preference given to class 1. We wish to minimize an objective function that is a convex combination of strictly increasing functions of two performance measures of interest related to the two classes.

More specifically, assume that the performance measures of class 1 and class 2 are denoted by $\bar{w}_1(\beta)$ and $\bar{w}_2(\beta)$, respectively. It is assumed that $\bar{w}_1(\beta)$ ($\bar{w}_2(\beta)$) decreases when the performance of class 1 (2) improves. Note the dependence of the performance measures on β. The assumption of a work-conserving and parametrized scheduling policy is represented by the following key properties:

Property (1) Function $\bar{w}_1(\beta) + \bar{w}_2(\beta)$ is independent of β.

Property (2) Functions $\bar{w}_2(\beta)$ and $\bar{w}_1(\beta)$ are continuous and strictly monotonic (increasing and decreasing, respectively) w.r.t. β on the interval $[0, 1]$.

While the first property is the formalization of the work-conserving nature of the scheduling policy, the second property characterizes the meaning of parameter β, i.e., the degree of preference given to class 1. The objective function to be minimized is given by (2.11) which we repeat here for completeness

$$F(\beta, \gamma) \triangleq \gamma g_1(\bar{w}_1(\beta)) + (1-\gamma) g_2(\bar{w}_2(\beta)), \tag{2.27}$$

with $0 \leq \gamma \leq 1$. The functions g_1 and g_2 are analytic and increasing functions. From the continuity of $\bar{w}_j(\beta)$ (Property 2) and continuity of g_j it is clear that $F(\beta, \gamma)$ is also continuous.

We emphasize that $\bar{w}_j$ here not necessarily denotes the mean unfinished work of class j. It is just a notation for a generic performance measure that satisfies both specified properties. Instead of average values of stationary variables, some transient (discounted) costs (over a finite or infinite time horizon) can also be used as performance measures [13, 22, 33]. For instance, the discounted holding cost of class j over an infinite time horizon is defined as

$$\bar{w}_j(\beta) = \mathrm{E}\left[\int_0^\infty e^{-\chi t} u_j(t, \beta) dt\right], \tag{2.28}$$

with $u_j(t, \beta)$ the buffer occupancy of class j at time t and χ the discount factor. Since these performance measures satisfy the two properties, we can use the results of this chapter for objective functions in these measures as well.

With these two properties as prerequisites, Theorems 2.2-2.4 can be reproven and rewritten in this setting. We will leave the proof to the reader as it is basically identical to the proofs of the corresponding theorems in the previous section. For completeness, we do reformulate the theorems below, i.e. omitting references to the specific GPS case. $\phi(\beta)$ is defined exactly like in (2.12), Lemma 2.3 is valid without changes.

Theorem 2.7. *Assume $g'_j(x) > 0$ and $g''_j(x) = 0$ for $j = 1, 2$ and all x in the region of interest (i.e., the functions g_j are linearly increasing in x). Then $\phi(\beta) = \phi_C$ is a constant function. As a result, either $\beta = 0$ or $\beta = 1$ is optimal, namely*

(i) for $\gamma < \phi_C$, $\beta = 0$ is optimal,

(ii) for $\gamma > \phi_C$, $\beta = 1$ is optimal, and

(iii) for $\gamma = \phi_C$, all values of β are equally optimal.

Theorem 2.8. *Assume $g'_j(x) > 0$ and $g''_j(x) \geq 0$ for $j = 1, 2$ and $g''_j(x) > 0$ for at least one j, for all x in the region of interest (i.e., the functions g_j are strictly increasing and convex in x with at least one of them strictly convex). Then $\phi(\beta)$ is a strictly increasing function. As a result,*

(i) for $\gamma \leq \phi(0)$, $\beta = 0$ is optimal,

(ii) for $\gamma \geq \phi(1)$, $\beta = 1$ is optimal, and

(iii) for $\phi(0) < \gamma < \phi(1)$, $\beta_{opt}(\gamma) = \phi^{-1}(\gamma)$ is optimal, with ϕ^{-1} the inverse function of ϕ. The function $\beta_{opt}(\gamma)$ has the following properties: it is different from 0 and 1 and it is strictly increasing.

Theorem 2.9. *Assume $g'_j(x) > 0$ and $g''_j(x) \leq 0$ for $j = 1, 2$ and $g''_j(x) < 0$ for at least one j, for all x in the region of interest (i.e., the functions g_j are strictly increasing and concave in x with at least one of them strictly concave). Then $\phi(\beta)$ is a strictly decreasing function. As a result, either $\beta = 0$ or $\beta = 1$ is optimal, namely*

(i) for $\gamma \leq \phi(1)$, $\beta = 0$ is optimal,

(ii) for $\gamma \geq \phi(0)$, $\beta = 1$ is optimal, and

(iii) for $\phi(1) < \gamma < \phi(0)$, $\beta = 0$ ($\beta = 1$) is optimal if $F(0, \gamma) <$ $(>)\, F(1, \gamma)$; if $F(0, \gamma) = F(1, \gamma)$, both $\beta = 0$ and $\beta = 1$ are optimal.

2.3.2 Some illustrative examples

Discrete-time GPS

A first example is obviously discrete-time GPS as we discussed in Section 2.2.2 in full detail. In [134], we studied discrete-time GPS with service times general and class-dependent but independent from customer to customer, i.e. no longer a deterministic single slot as in Section 2.2.2. The results are interesting but not fundamentally different from the ones in Section 2.2.2. They add little extra value here, so we omit them and refer the interested reader to [134].

A coupled processor model

Assume the following continuous-time coupled-processor model, see also [58]. Two types of tasks arrive to the system according to two independent Poisson arrival processes and are backlogged in two different queues. Service times are exponentially distributed and the service rate is equal to μ. When one of the queues is empty, the complete service rate is allocated to the other queue. When both queues are non-empty, the service rate μ is split, queue 1 getting a service rate of $\beta\mu$ and queue 2 of $(1-\beta)\mu$. Assume that the performance measure $\bar{w}_j(\beta)$ of queue j is the mean stationary unfinished work in queue j. Then both properties apply: (1) the scheduling is work-conserving meaning that the total (mean) unfinished work is independent of β, and (2) a higher value of β gives more preference to queue 1 leading to a strictly increasing $\bar{w}_2(\beta)$. Hence, this example fits our framework. We omit further discussion of this famous model.

Relation of the framework to the $c\mu$-rule

The $c\mu$-rule is well known in literature [13, 33, 44, 71, 98]. With μ_j defined as the service rate for queue j and c_j the coefficient of queue j in the linear cost function, it says that strict priority scheduling with the highest priority for the queue with the highest $c_j\mu_j$ is optimal in the case of N queues sharing a single resource. This has been proven for a wide range of cost functions containing a linear combination of some queue size measure. For instance in [33] the authors prove the optimality for N classes of tasks, each requiring a service time of a number of slots generated by a geometric distribution with mean μ_j^{-1}. The service discipline is preemptive, meaning the server can interrupt the service of a task to continue later. The authors minimize the expected total discounted waiting cost

$$E\left[\sum_{t=1}^{T}\chi^t\sum_{j=1}^{N}c_jU_j^{\pi}(t)\right], \tag{2.29}$$

with χ the discounting parameter and $U_j^{\pi}(t)$ the queue content of queue j at time t under scheduling policy π. This minimization is over all possible scheduling policies π that only depend on previous and present queue lengths and control actions.

In this paragraph, we show that the conclusions made there also fit our framework in case of two queues. To this end, we rewrite the cost function for two queues as:

$$E\left[\sum_{t=1}^{T}\chi^t\left(c_1U_1^\pi(t)+c_2U_2^\pi(t)\right)\right] \tag{2.30}$$

$$= c_1E\left[\sum_{t=1}^{T}\chi^tU_1^\pi(t)\right]+c_2E\left[\sum_{t=1}^{T}\chi^tU_2^\pi(t)\right] \tag{2.31}$$

$$= c_1\mu_1\sum_{t=1}^{T}\chi^tE\left[W_1^\pi(t)\right]+c_2\mu_2\sum_{t=1}^{T}\chi^tE\left[W_2^\pi(t)\right], \tag{2.32}$$

where $W_j^\pi(t)$ denotes the unfinished work in queue j. In the last step, we used that $E\left[W_j^\pi(t)\right] = \mu_j^{-1}E\left[U_j^\pi(t)\right]$ as each packet in the queue on average accounts for a service time of μ_j^{-1}. The first packet in the queue could have been serviced for some time before, so we should only account for a residual service time; however, as the service times are geometrically distributed so are the residual service times. This cost function is of the form of Equation (2.27), by substituting $\gamma = \frac{c_1}{c_1+c_2}$, $g_j(x) = (c_1+c_2)\mu_jx$ and $\bar{w}_j(\beta) = \bar{w}_j(\beta(\pi)) = \sum_{t=1}^{T}\chi^tE[W_j^\pi(t)]$. The β parameter is introduced by ordering all policies π according to increasing values for $\sum_{t=1}^{T}\chi^tE[W_2^\pi(t)]$ and mapping each policy π to a specific value $\beta(\pi)$ in the interval $[0,1]$, so that increasing $\beta(\pi)$ leads to increasing $\sum_{t=1}^{T}\chi^tE[W_2^\pi(t)]$. As a result, a continuum is reached as every $\bar{w}_j(\beta(\pi))$ can be obtained by using different strict priority policies in different busy periods (as is done in e.g. [61]). For two distinct policies π_1 and π_2, $\sum_{t=1}^{T}\chi^tE[W_2^{\pi_1}(t)] = \sum_{t=1}^{T}\chi^tE[W_2^{\pi_2}(t)]$ iff $\beta(\pi_1)=\beta(\pi_2)$. Furthermore, we discard all non work-conserving policies (these will never lead to an optimal solution in this preemptive setting), as a consequence $\sum_{t=1}^{T}\chi^tE[W_1^\pi(t)]+\sum_{t=1}^{T}\chi^tE[W_2^\pi(t)] = \sum_{t=1}^{T}\chi^tE[W_T(t)]$ is a constant independent of the scheduling policy. Property 1 is thus fulfilled. Property 2 is also fulfilled, as by construction $\bar{w}_2(\beta(\pi))$ is strictly increasing and as a consequence of property 1 $\bar{w}_1(\beta(\pi))$ is strictly decreasing.

For this configuration, we get $\phi_C = \frac{\mu_2}{\mu_1+\mu_2}$, see Equation (2.12). Theorem 2.7 then states that $\beta = 0$ is optimal when $\gamma < \phi_C$ with $\gamma = \frac{c_1}{c_1+c_2}$; this is when $c_1\mu_1 < c_2\mu_2$. $\beta = 1$ is optimal when $\gamma > \phi_C$, i.e., when $c_1\mu_1 > c_2\mu_2$. $\beta = 0$ ($\beta = 1$) means that we should use the policy π that minimizes (maximizes) $\sum_{t=1}^{T} \chi^t E[W_2^\pi(t)]$, we should thus give strict priority to tasks of class 2 (class 1). Our framework thus also leads to the $c\mu$-rule, for the cases in which it is applicable. The framework can, for instance, also be used with the cost defined in Equation (2.28). As such, the $c\mu$-rule for two classes can be seen as a special case of our more general framework, more specifically a special case of Theorem 2.7 with linear g_j functions.

Semi-preemptive priority scheduling

The last example we discuss is a particular priority queueing system. Priority queueing systems are usually labeled as either preemptive or non-preemptive. In the first case, high-priority arrivals interrupt service of an on-going low-priority service (which is later resumed or repeated), while new high-priority arrivals have to wait for service at least until the end of the ongoing service in the non-preemptive systems. In that respect, preemptive priority delivers more priority than non-preemptive priority. However, one could wonder which one is better, or, even if there is nothing better *in between*. Indeed, non-preemptiveness seems to be reasonable when low-priority service times are small, while for long low-priority service times preemptive priority could be preferable. For this reason, semi-preemptive priority rules have been investigated in the recent past [87,142]. In [142], a (discrete) threshold T on the remaining service time of a low-priority service at an arrival instant of a high-priority task is introduced[6]. If the remaining service time is larger than or equal to T, the service is interrupted; otherwise, it is not interrupted. We assume interrupted services are resumed afterwards.

The idea is to optimize the system in the positive integer T. We stress that the problem is somewhat different from GPS. Here, we assume that tasks of class 1 have priority over tasks of class 2 inherently, and we only allow optimization of interruptions of low-priority services, through the threshold T. The parameter T can be transformed to play the role of β here. Assume, for instance, that service times of class 2 are bounded by the positive integer S ($S > 1$)[7]. Then T is between 1 and S, and by defining β as $(S-T)/(S-1)$ and $\bar{w}_j(\beta)$ as the average stationary unfinished work of class j ($j = 1, 2$), the framework of Section 2.3.1 applies. Indeed, β is to be optimized in the interval $[0, 1]$ (0 corresponding to the non-preemptive policy

[6] We assume we know the exact value of the remaining service time at all times.

[7] If they are not bounded, one can still introduce a (large enough) S which would mean that for a remaining service time larger than this value, the class-2 service is interrupted anyway.

$T = S$ and 1 to the preemptive one $T = 1$), the scheduling is work-conserving (property 1 is valid), and more preference is given to class 1 if β increases (property 2 is valid).

In contrast with GPS, a queueing system with this kind of scheduling is analytically tractable, at least for some arrival and service processes. For instance, a discrete-time semi-preemptive priority queueing system with geometric service times for the high-priority class and fixed service times for the low-priority class, was analyzed, in [142]. We will use the formulas from this paper for optimization and relate them to our framework.

The numbers of task arrivals are independent and identically distributed from slot to slot, but correlation between the numbers of arrivals of both classes within one slot is allowed. The mean number of arrivals of class j equals λ_j. Service times of class 1 are geometrically distributed with mean $\bar{s}_1$ and service times of class 2 are deterministically equal to S. In [142], average values of the mean buffer occupancies and mean delays of both classes have been calculated. The two key properties of our framework, however, do not directly apply to these performance measures. Therefore, we assume the performance measures of the objective function (2.27) to be the average unfinished work of both classes[8]. The average unfinished work of both classes can be calculated from the results of [142]. We refer to the appendix of [134] for details. We find that

$$\begin{aligned}\bar{w}_1(\beta) =& \frac{\bar{s}_1[\bar{s}_1\mathrm{Var}[a_1] + \rho_1(1-\lambda_1)]}{2(1-\rho_1)} \\ &+ \frac{\rho_1\lambda_2(1-\beta)(S-1)[\beta+(1-\beta)S]}{2(1-\rho_1)}\end{aligned} \tag{2.33}$$

and

$$\begin{aligned}\bar{w}_2(\beta) = & \frac{\rho_2}{2} + \frac{S^2\mathrm{Var}[a_2] + 2\bar{s}_1 S\mathrm{Cov}[a_1,a_2]}{2(1-\rho_T)} \\ &+ \frac{\rho_2\bar{s}_1[\bar{s}_1\mathrm{Var}[a_1] + \lambda_1(\bar{s}_1-1)]}{2(1-\rho_T)(1-\rho_1)} \\ &- \frac{\rho_1\lambda_2(1-\beta)(S-1)[\beta+(1-\beta)S]}{2(1-\rho_1)},\end{aligned} \tag{2.34}$$

with ρ_T the total load and ρ_j the class-j arrival load (i.e., $\rho_1 \triangleq \lambda_1\bar{s}_1$ and $\rho_2 = \lambda_2 S$), $\mathrm{Var}[a_j]$ the variance of the number of class-j arrivals during a slot, and $\mathrm{Cov}[a_1, a_2]$ the covariance between the numbers

[8]We show how to deal with mean buffer occupancies and mean delays later.

of class-1 and class-2 arrivals. Summing the unfinished work of both classes leads to

$$\begin{aligned}\bar{w} =& \frac{\rho_1+\rho_2}{2} + \frac{\bar{s}_1^2\mathrm{Var}[a_1] + 2\bar{s}_1 S\mathrm{Cov}[a_1,a_2] + S^2\mathrm{Var}[a_2]}{2(1-\rho_1-\rho_2)} \\ &+ \frac{\rho_1(\bar{s}_1-1)}{2(1-\rho_1-\rho_2)}.\end{aligned}$$

It is clear that $\bar{w}_1(\beta)$ and $\bar{w}_2(\beta)$ fulfill the two properties, so our framework applies. This means, for instance, that for weighted combinations of linear or concave functions of the mean unfinished work of both classes, either preemptive or non-preemptive priority is optimal. For convex functions of the mean unfinished work, a threshold T different from 1 or S can be optimal. We note that T is a discrete parameter, while the optimization will deliver a continuous optimal β. However, since we know that the objective function is unimodal in β, the optimal discrete threshold will be one of the two nearest discrete values.

We conclude this paragraph with an example where a particular linear objective function does *not* lead to an optimal value for β of 0 or 1. Consider following objective function

$$F(\beta,\gamma) = \gamma\bar{u}_1(\beta) + (1-\gamma)\bar{u}_2(\beta), \tag{2.35}$$

with $\bar{u}_j(\beta)$ the mean buffer occupancy. It was shown in [142] that the optimal threshold T (equivalent to the optimal β, see above) is not necessarily 1 or S but can be any value in between. The reason is obviously that $\bar{u}_1(\beta)$ and $\bar{u}_2(\beta)$ do not fulfill both properties, viz., Property 1. However, we can still use our framework to prove that the optimal threshold is not necessarily at the extremes. Let us transform objective function (2.35) into an objective function in the mean unfinished work of both classes, i.e., into

$$F(\beta,\gamma) = \gamma g_1(\bar{w}_1(\beta)) + (1-\gamma)g_2(\bar{w}_2(\beta)). \tag{2.36}$$

In order to find the functions g_j, we need to investigate the relations between the mean buffer occupancies and the mean unfinished work of both classes. For class 1, this relation is immediate from the memoryless property of the class-1 service times

$$\bar{u}_1(\beta) = \frac{\bar{w}_1(\beta)}{\bar{s}_1},$$

and we thus have a linear relation. For class 2, a relation is harder to find. From calculations in the appendix of [134], we obtain that

$$\bar{u}_2(\beta) = \frac{\bar{w}_2(\beta)}{S} + \frac{\lambda_2(S-1)}{2} + \frac{\lambda_2\rho_1\beta(S-1)(1-\beta+\beta S)}{2S(1-\rho_1)}. \tag{2.37}$$

This does not provide a direct relation g_2 between $\bar{u}_2(\beta)$ and $\bar{w}_2(\beta)$, as the right-hand side of (2.37) also explicitly depends on β. Since $\bar{w}_2(\beta)$ is strictly increasing in β, however, this function can be inverted, and β can be written as function of $\bar{w}_2$. So, a g_2 exists. We will not actually calculate g_2, but calculate its two first derivatives (to see how it fits the framework). The first derivative of g_2 can be found from (2.37) as follows:

$$\begin{aligned}\frac{dg_2(\bar{w}_2(\beta))}{d\beta} =& g_2'(\bar{w}_2(\beta)) \cdot \bar{w}_2'(\beta) \\ =& \frac{\bar{w}_2'(\beta)}{S} + \frac{\lambda_2 \rho_1 (S-1)[1+2\beta(S-1)]}{2S(1-\rho_1)}.\end{aligned}$$

This leads to

$$g_2'(\bar{w}_2(\beta)) = \frac{1}{S} + \frac{\lambda_2 \rho_1 (S-1)[1+2\beta(S-1)]}{2S(1-\rho_1)\bar{w}_2'(\beta)},$$

which is strictly positive as all factors in both terms are positive. Then taking the derivative to β once more yields

$$\begin{aligned}g_2''(\bar{w}_2(\beta)) = \frac{\lambda_2 \rho_1 (S-1)}{2S(1-\rho_1)(\bar{w}_2'(\beta))^2} \cdot \Big(& 2(S-1)\bar{w}_2'(\beta) \\ & - [1+2\beta(S-1)]\bar{w}_2''(\beta)\Big).\end{aligned}$$

Since the coefficient of the quadratic term β^2 of $\bar{w}_2(\beta)$ is smaller than zero (see expression (2.34)), $\bar{w}_2(\beta)$ is concave, and as a result g_2 is convex. This explains why an optimal T different from the endpoints was found in [142]. In fact, we can calculate $\phi(0)$ and $\phi(1)$, resulting in

$$\begin{aligned}\phi(0) =& \frac{2\bar{s}_1}{2(\bar{s}_1 + S) - 1}, \\ \phi(1) =& \frac{2\bar{s}_1}{2\bar{s}_1 + 1}.\end{aligned} \tag{2.38}$$

For $\gamma \in]\phi(0), \phi(1)[$, $T = 1$ or $T = S$ are not necessarily[9] optimal.

Note that $\phi(0)$ and $\phi(1)$ depend on the (mean) service times of both classes only. In fact, $\phi(1)$ depends only on the mean service time of class 1, i.e., $\bar{s}_1$. Since $\bar{s}_1 \in [1, \infty[$, the range for $\phi(1)$ is $[2/3, 1[$. This is also plotted in Figure 2.18, where we show $\phi(1)$ as a function of $\bar{s}_1$. In the same figure, $\phi(0)$ is also presented as a function of $\bar{s}_1$ for several

[9] Due to the discretization it is possible for $T = 1$ or $T = S$ to still be optimal in the boundary cases.

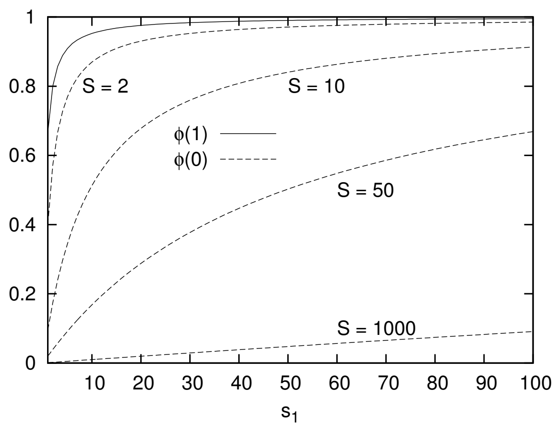


Figure 2.18: The behavior of $\phi(0)$ and $\phi(1)$ as a function of S and $\bar{s}_1$

values of the fixed service time of class 2, i.e., S. For a given value of $\bar{s}_1$, $\phi(0)$ consequently ranges from $\phi(1)$ to 0. Summarized, since $S \in [1, \infty[$ it is clear that for increasing service times of one of the classes (or both simultaneously), the interval $[\phi(0), \phi(1)]$ increases. Thus the range of values for γ ($\gamma \in]\phi(0), \phi(1)[$), where the optimum of the objective function F is different from the endpoints increases.

The problem described in this example is an interruption problem rather than a scheduling problem. In this kind of problems, the question of what type to schedule when the server becomes free is assumed to be answered, and the remaining question is an optimal interruption policy. We showed that this problem also fits our framework. This example also demonstrates that linear objective functions in the mean buffer occupancies can result in convex objective functions in the mean unfinished work. Therefore, even for linear objective functions, the optimal value of the parameter is not necessarily one of the endpoints. It would be interesting to investigate if this convexity holds for more general arrival and service time models.

2.4 Summary

In this chapter, we have shown what performance region is achievable by a discrete-time GPS system and proved the monotonicity of the performance measures. This enabled us to develop important theorems about the optimization of the system. These theorems make it possible to, a priori, determine the behavior of the objective function and in what cases strict priority is optimal and in which cases *true GPS*. This significantly speeds up or cancels any computationally expensive optimization search. For the cases wherein this search is needed anyway, we presented and evaluated some algorithms. Lastly, we have shown that part of the results presented here can be ported to a wider class of scheduling problems of which we presented some examples. The results from this chapter have also been published in [133, 134, 138, 146].

"Hoge klemmers, verre zwemmers en schaatsers op het ijs, ze zijn allemaal niet goed wijs."
— Jerome Vanlerberghe

3 Three Classes

3.1 Extending discrete-time GPS to three classes

Extending the discrete-time two-class version of GPS, introduced in Section 1.3, to a version with more classes is not as easy as it seems. For instance, if we envision the version from Figure 3.1 with 3 classes, we lose key properties of the two-class model. This makes that analysis methods for the two-class model do not extend to the three-class version.

In the version of Figure 3.1, when all queues are backlogged, classes 1, 2 and 3 are served with probabilities $\beta_1, \beta_2, 1 - \beta_1 - \beta_2$, respectively. If only queue 1 is empty, its probability share is propor-

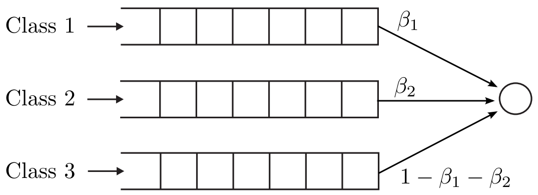


Figure 3.1: Figure of model with 3 classes

tionally redivided amongst the non-empty classes 2 and 3, i.e. class 2 (class 3) is served with probability $\frac{\beta_2}{1-\beta_1}$ $(\frac{1-\beta_1-\beta_2}{1-\beta_1})$. The policy is analogous if only queue 2 or 3 are empty. Lastly, when two queues are simultaneously empty, the only remaining non-empty queue is served.

The special case $\beta_1 = 1$ gives strict priority to class-1 customers. With $\beta_1 = 1$ the weights for queues 2 and 3 are 0 and, as a result, it is undefined what to do when queue 1 is empty. Give priority to queue 2 or 3 is an option, which produces a strict priority case, but anything in between is also possible. As shown, the definition of this three-class policy is non-trivial. To keep all priority policies as a special case in the three-class version, extra parameters other than β_1 and β_2 are needed. Another possibility is to add the restriction that no queue weight can equal 1. That way the priority cases are no longer possible but can be viewed as limit cases. For instance the priority case $1 > 2 > 3$ is obtained by $\beta_1 \to 1$, $\beta_2 \to 0$ and $1 - \beta_1 - \beta_2 = 0$.

The main disadvantage however is the difficult analysis of its performance and the subsequent optimization of the parameters. Obtaining these β_1 and β_2 that lead to a desired performance vector $\bar{\boldsymbol{w}}^*$ is far from straightforward. The fact that a small perturbation in one of the β_i $(i = 1, 2)$ probabilities has an influence on each of the $\bar{w}_j$ $(j = 1, 2, 3)$, makes for instance hierarchical optimization of the β_i's cumbersome.

To mitigate these analysis and optimization challenges of GPS, we investigate a hierarchical version of the aforementioned discrete-time GPS. This version is related to other practical implementations of the idealized H-GPS scheduling discipline, as for instance described in [15,35,66,82]. In this discrete-time H-GPS version, we introduce two hierarchical levels. In each slot, assuming all queues are backlogged, the server first decides on the first hierarchical level to either serve queue 1 with probability β_1, or delegate service to hierarchical level 2 with probability $1 - \beta_1$. If service is delegated to level 2, queue 2 is served with probability β_2 and queue 3 is served with probability $1 - \beta_2$. If queue 1 is empty, service is by default delegated to level 2. Conversely, if queues 2 and 3 are empty queue 1 is served. If service is delegated to level 2 and one of the queues on that level is empty, the other queue is served. Obviously no service happens when the system is empty. The policy is illustrated in Figure 3.2. It is clear from this definition that β_2 has no influence on the decision whether or not to serve queue 1. The latter decision is taken on the first hierarchical level where β_2 plays no role. As a consequence, β_2 has no influence on $\bar{w}_1$, and thus β_1 and β_2 can be optimized hierarchically.

For completeness of the description of the dynamics of the H-

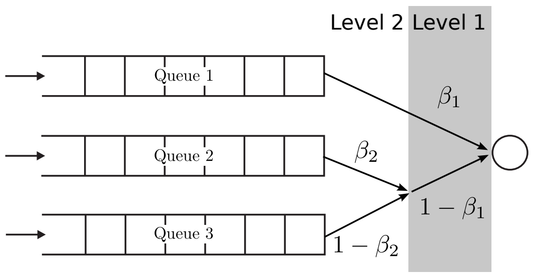


Figure 3.2: Discrete-time Hierarchical Generalized Processor Sharing

GPS system, we specify the system equations below using the usual symbols defined in Section 1.4.

- if $\boldsymbol{w}_k = \boldsymbol{0}$

$$\boldsymbol{w}_{k+1} = \boldsymbol{a}_k$$

- if $w_{i,k} > 0$; $w_{j,k} = 0$ with $j = \{1,2,3\} \setminus \{i\}$

$$\begin{aligned} w_{i,k+1} &= w_{i,k} - 1 + a_{i,k} \\ w_{j,k+1} &= a_{j,k} \end{aligned}$$

- if $w_{1,k} = 0$; $w_{2,k}, w_{3,k} > 0$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,1,0) + \boldsymbol{a}_k \quad \text{w.p.} \quad \beta_2 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,0,1) + \boldsymbol{a}_k \quad \text{w.p.} \quad 1-\beta_2 \end{aligned}$$

- if $w_{2,k} = 0$; $w_{1,k}, w_{3,k} > 0$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (1,0,0) + \boldsymbol{a}_k \quad \text{w.p.} \quad \beta_1 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,0,1) + \boldsymbol{a}_k \quad \text{w.p.} \quad 1-\beta_1 \end{aligned}$$

- if $w_{3,k} = 0$; $w_{1,k}, w_{2,k} > 0$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (1,0,0) + \boldsymbol{a}_k \quad \text{w.p.} \quad \beta_1 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,1,0) + \boldsymbol{a}_k \quad \text{w.p.} \quad 1-\beta_1 \end{aligned}$$

- if $w_{j,k} > 0$ with $j = \{1,2,3\}$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (1,0,0) + \boldsymbol{a}_k \quad \text{w.p.} \quad \beta_1 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,1,0) + \boldsymbol{a}_k \quad \text{w.p.} \quad (1-\beta_1)\beta_2 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,0,1) + \boldsymbol{a}_k \quad \text{w.p.} \quad (1-\beta_1)(1-\beta_2) \end{aligned}$$

However, it is unclear whether the achievable performance region of H-GPS matches the one from GPS, i.e., we are not certain that each vector $\bar{\boldsymbol{w}} = (\bar{w}_1, \bar{w}_2, \bar{w}_3)$ that can be generated by a particular GPS system can also be constructed with our new H-GPS system (and vice versa). Therefore, we first study the performance region of H-GPS and GPS in the next section. We show that a single H-GPS system cannot achieve the performance region of GPS. There are however several possibilities to route classes to queues, delivering different performance for the classes. We prove that with the union of these different H-GPS configurations, the whole GPS performance region is achieved. Subsequently in Section 3.3, we look at the optimization of discrete-time H-GPS. Lastly, in Section 3.4, we investigate the extension to more than three classes.

3.2 Achievable Region

We start by identifying the achievable region of strict priority scheduling policies. As in Section 2.1, we express an achievable performance point as a vector $\bar{\boldsymbol{w}} = (\bar{w}_1, \bar{w}_2, \bar{w}_3)$. For strict priority, it is clear that only $3! = 6$ different priority orderings exist and that we thus have 6 achievable performance vectors.

One can intuitively see that, if we only consider work-conserving policies, the performance vector of every policy will always be included in the convex polytope with the 6 performance vectors of strict priority as vertices. This is also clarified in for instance [21, 45, 70, 72, 112, 120]. Furthermore, Federgruen and Groenevelt [61] have shown that every performance vector in this polytope can be achieved by a certain policy. In their paper, they construct a policy whereby in each busy cycle a different priority ordering is used. By selecting the different orderings with appropriate probabilities in the subsequent busy cycles, the policy achieves the requested performance vector in the limit (averaged over all busy cycles). In practice, this policy is not very valuable as the performance for customers of a certain class strongly depends on the busy cycle of their arrival. This is an undesired property for practitioners; this policy lacks consistency of performance in time. It mainly has merit from a theoretical point of view, for proving the feasibility of any point in the polytope by at least one scheduling policy.

Let us first look at this polytope. We denote the area enclosed by the polytope by $\Omega \subset \mathbb{R}^3$; the performance vector resulting from a certain work-conserving policy will always be an element of Ω. An example of such a polytope Ω is drawn with a thick line in Figure 3.3.

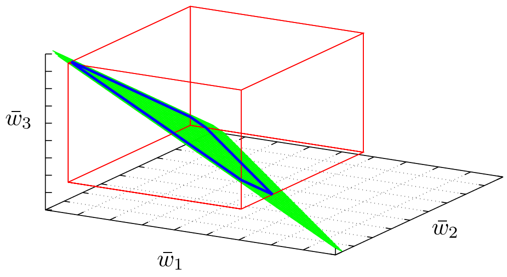


Figure 3.3: Enclosed by thick lines: polytope enclosing possible performance vectors for work-conserving policies in 3D

As can be seen from the figure, Ω lies in a plane. The reason is that, given that we have three classes and only consider work-conserving policies, the mean total amount of work in the system is a constant. This constant is:

$$\bar{w}_T = \bar{w}_1 + \bar{w}_2 + \bar{w}_3, \tag{3.1}$$

whereby $\bar{w}_i$ depends on the policy and $\bar{w}_T$ is a constant, independent of the policy. Expression (3.1) is the equation for the plane shown in Figure 3.3. As such, we can draw the polytope in 2D for clarity, discarding $\bar{w}_3$ as it can be calculated from $\bar{w}_1$ and $\bar{w}_2$. An example is shown in Figure 3.4. In this figure the black circles with notation $(A > B > C)$ (with (A, B, C) a permutation of $(1, 2, 3)$) designate the performance vector where class A has highest priority, followed by class B and class C has the lowest priority.

The discrete-time version of GPS can achieve any performance vector inside Ω, by choosing the weights β_1 and β_2 appropriately[1]. The border of the polytope (and the vertices that correspond to priority scheduling) is reached in the limit where one of the probabilities is 1 (high priority) and both others 0. In this limit case however, it is necessary to further specify the bandwidth shares between the classes with 0 probability (as mentioned in the previous section), as it is otherwise unclear which queue to serve in the event that the

[1] We do not have a formal proof for this claim nor did we find any in the literature. However, we intuitively feel from the GPS mechanism and the continuous nature of the parameters together with the fact that the limiting cases constitute the priority vertices, that this is indeed true. Extensive simulations that we did confirmed this belief.

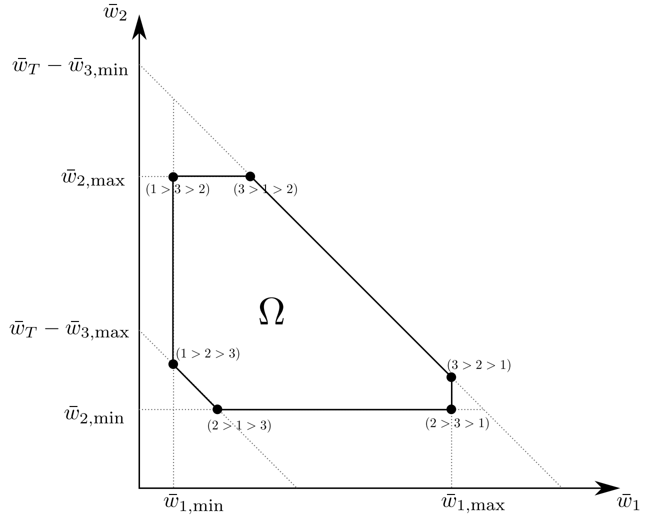


Figure 3.4: Polytope enclosing possible performance vectors for work-conserving policies in 2D

high-priority queue is empty.

Reasoning about the H-GPS system, it is intuitively clear that priority ordering $(1 > 2 > 3)$ is achieved by choosing $\beta_1 = 1$ and $\beta_2 = 1$. This is easily done for $(1 > 3 > 2)$, $(3 > 2 > 1)$ and $(2 > 3 > 1)$ as well, choosing respectively $(\beta_1 = 1, \beta_2 = 0)$, $(\beta_1 = 0, \beta_2 = 0)$ and $(\beta_1 = 0, \beta_2 = 1)$. However, it is also clear that it is impossible to achieve $(3 > 1 > 2)$ and $(2 > 1 > 3)$ with this H-GPS system. For instance to assign class 3 with the highest priority, $\beta_1 = 0$ and $\beta_2 = 0$ is needed, which however results necessarily (due to the hierarchy) in class 2 having priority over class 1. Subsequently, this means that we cannot achieve the full polytope Ω using this H-GPS system. However, we can effectuate $(3 > 1 > 2)$ by switching hierarchy levels, i.e., by sending class 3 customers to queue 1 (Q1), class 1 customers to queue 2 (Q2), class 2 customers to queue 3 (Q3) and by choosing $\beta_1 = 1; \beta_2 = 1$. If we schematically represent the three-class H-GPS system as: (Q1,(Q2,Q3)) then we can note the aforementioned system as: $(3, (1, 2))$. Effectively, for three classes we have 3 possible H-GPS systems (omitting symmetric systems, interchanging the classes on the deepest level): $(1, (2, 3))$, $(2, (1, 3))$ and $(3, (1, 2))$. The aim of this section is to prove the next theorem that states that the union of the performance regions of these 3 systems is indeed equal to the entire polytope Ω.

Theorem 3.1. *Every performance vector $\bar{\boldsymbol{w}}$ in Ω is feasible with at least one of the three possible H-GPS systems* $(1,(2,3))$, $(2,(1,3))$ *or* $(3,(1,2))$.

The remainder of this section is dedicated to proving this crucial theorem. We start with an example and a sketch of the proof to provide some intuition. Subsequently, we state two lemmas used in the formal proof of Theorem 3.1 at the end of this section.

As an example, Figure 3.5 shows the performance regions of the three possible H-GPS systems for a specific arrival pattern (i.e.: a multinomial arrival process with $\lambda_T = 0.9, N = 16, \alpha_1 = 0.16, \alpha_2 = 0.32$). From the figure we can, for instance, see that indeed $(1,(2,3))$ does not achieve the performance of priority ordering $(3 > 1 > 2)$ and $(2 > 1 > 3)$. We also see that the performance vectors in the regions that are outside of the feasible region for H-GPS $(1,(2,3))$, are inside the feasible region of either $(2,(1,3))$ or $(3,(1,2))$.

First, we observe that given a certain performance vector $\bar{\boldsymbol{w}}^* = (\bar{w}_1^*, \bar{w}_2^*, \bar{w}_3^*)$ in the polytope Ω, each of the three H-GPS systems can at least provide a performance vector for which the performance of the class on its highest level is correct. For instance, for H-GPS $(2,(1,3))$ a b_2 exists for which $\bar{w}_2^* = \bar{w}_2^{(2,(1,3))}(b_2, b_2')$. This is evidently the case as $\bar{w}_2^{(2,(1,3))}(b_2, b_2')$ spans from $\bar{w}_{2,\max}$ to $\bar{w}_{2,\min}$ as b_2 traverses from 0 to 1 (independently of b_2'). This observation forms the basis of our proof. As such $\exists b_1, b_2, b_3 : \bar{w}_1^* = \bar{w}_1^{(1,(2,3))}(b_1, \cdot), \bar{w}_2^* = \bar{w}_2^{(2,(1,3))}(b_2, \cdot), \bar{w}_3^* = \bar{w}_3^{(3,(1,2))}(b_3, \cdot)$. It is however unsure (and not always the case) that a b_2' can be found such that $\bar{w}_1^* = \bar{w}_1^{(2,(1,3))}(b_2, b_2')$ and $\bar{w}_3^* = \bar{w}_3^{(2,(1,3))}(b_2, b_2')$. In the formal proof of Theorem 3.1, we show that the assumption that $\bar{\boldsymbol{w}}^*$ cannot be achieved by *any* of the systems leads to a contradiction with regards to the b_i values. The two lemmas stated and explained in the following subsections, help us establishing the behavior of the performance metrics with regards to their parameter settings.

The proofs presented in this paper are based on coupling arguments, as in Section 2.1. For the proofs (and thus the theorems) we impose some assumptions on the arrival process. We assume the arrival process is such that $\boldsymbol{w}_k$ is a discrete-time regenerative process and that there are arrivals for each of the three classes. Additionally, we assume (i) the limiting distribution of $\boldsymbol{w}_k$ exists and has a finite mean and (ii) the unfinished work in all queues at regeneration points does not depend on the (configuration of the) specific policy. In practice this means that a finite-mean regeneration cycle length

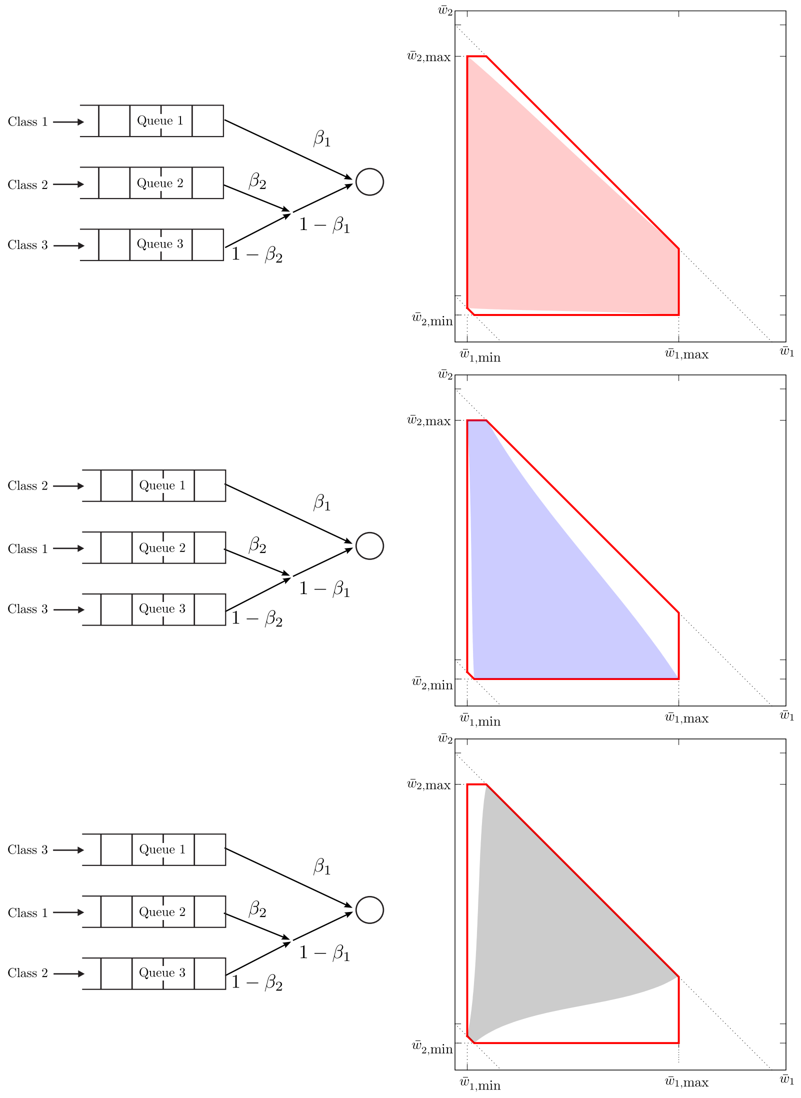


Figure 3.5: Performance space (grey area) for three different H-GPS systems with 3 classes of customers: $(1,(2,3))$-top ; $(2,(1,3))$-middle and $(3,(1,2))$-bottom

and empty system ($\boldsymbol{w}_k = \mathbf{0}$) at regeneration epochs suffice as (not very restrictive) conditions [7].

3.2.1 The influence of β_1 in H-GPS

Lemma 3.1. *For* (A, B, C) *an arbitrary permutation of* $(1, 2, 3)$*, we have for each* β *and* $\Delta\beta$ *such that* $0 \leq \beta < \beta + \Delta\beta \leq 1$*:*

(i) $\bar{w}_A^{(A,(B,C))}(\beta, 1) \geq \bar{w}_A^{(A,(B,C))}(\beta + \Delta\beta, 1)$ *and*

(ii) $\bar{w}_C^{(A,(B,C))}(\beta, 1) \leq \bar{w}_C^{(A,(B,C))}(\beta + \Delta\beta, 1)$.

We formulate the lemma in terms of H-GPS $(A, (B, C))$ to emphasize, that it is generic for the three possible systems $(1, (2, 3))$, $(2, (1, 3))$ and $(3, (1, 2)$. Basically, Lemma 3.1 says that if we give class B priority over class C (second configuration parameter $\beta_2 = 1$), the mean unfinished work of class A (class C) will be decreasing (increasing) for increasing (first configuration parameter) β_1.

The proof of this lemma is based on a coupling argument. We couple arrivals and server decisions of two systems $S^{(A,(B,C))}(\beta, 1)$ and $S^{(A,(B,C))}(\beta + \Delta\beta, 1)$ and compare the sample paths of their unfinished work. As such, if $\Delta\beta$ would be equal to zero, the unfinished work would be equal in each queue in each slot. For making the decision which queue to serve, the server generates two decision variables $r_{1,k}$ and $r_{2,k}$ uniformly in the interval $[0, 1]$ in each slot k. Subsequently, these values are compared to β_1 and β_2 (i.e.: the weight parameters on the first and second hierarchical level, see Figure 3.2) to make a decision on which queue to serve. For instance, when all queues are backlogged in slot k, queue 2 will be served when $r_{1,k} \in [\beta_1, 1]$ and $r_{2,k} \in [0, \beta_2[$. Consequently, the sample paths for these two coupled systems can only start to diverge in a slot where $\beta \leq r_{1,k} < \beta + \Delta\beta$. We prove that there are always at least as much class-A (class-C) customers in $S^{(A,(B,C))}(\beta, 1)$ $(S^{(A,(B,C))}(\beta + \Delta\beta, 1))$ as in $S^{(A,(B,C))}(\beta + \Delta\beta, 1)$ $(S^{(A,(B,C))}(\beta, 1))$ sample path wise. To this end, we study the difference vector $\boldsymbol{\Delta w}_k = \boldsymbol{w}^{(A,(B,C))}(\beta + \Delta\beta, 1) - \boldsymbol{w}^{(A,(B,C))}(\beta, 1)$ from slot to slot. The proof then boils down to proving that the first (last) element of the difference vector is always negative (positive). The introduction of the difference vector is a concise way of comparing the sample paths of both coupled systems.

We show an illustrative example for the proof of Lemma 3.1 in Figure 3.6. In the first two slots of the example both systems are aligned. In slot 2, $\beta \leq r_{1,k} < \beta + \Delta\beta$ thus $S^{(A,(B,C))}(\beta, 1)$ serves class C (class B empty) and $S^{(A,(B,C))}(\beta + \Delta\beta, 1)$ serves class A.

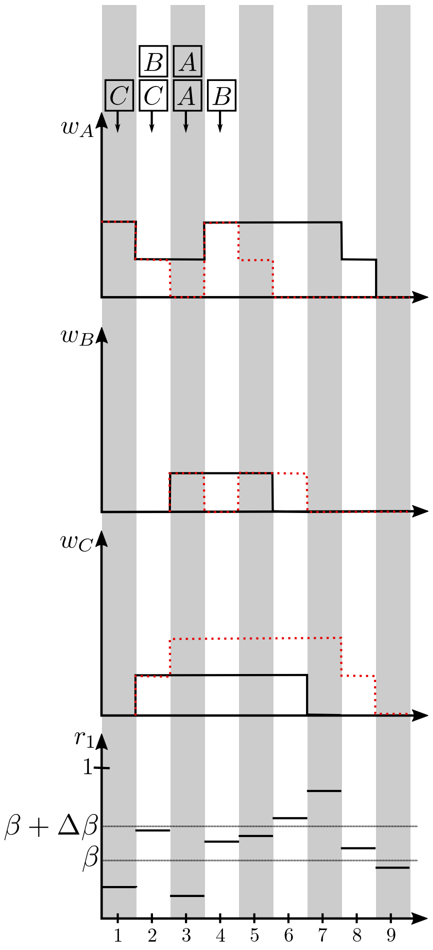


Figure 3.6: Illustration of Lemma 3.1 $S^{(A,(B,C))}(\beta + \Delta\beta, 1)$ (dotted) $S^{(A,(B,C))}(\beta, 1)$ (solid)

As a result, the difference vector in slot 3 is $(-1, 0, 1)$. In slot 3, $S^{(A,(B,C))}(\beta, 1)$ serves class A, but $S^{(A,(B,C))}(\beta + \Delta\beta, 1)$ serves class B as in that system class A is empty. Subsequently, the difference vector in slot 4 is $(0, -1, 1)$ and evolves back to $(-1, 0, 1)$ in slot 5. In slot 5, $\beta \le r_{1,k} < \beta + \Delta\beta$ thus $S^{(A,(B,C))}(\beta, 1)$ serves class B and $S^{(A,(B,C))}(\beta + \Delta\beta, 1)$ serves class A. In the next slot, the difference vector is thus $(-2, 1, 1)$, afterwards both systems empty at which point both systems synchronize. From this example, we see that it is possible for the second element of the difference vector to be positive as well as negative. As such, $\bar{w}_B^{(A,(B,C))}(\beta, 1)$ is not necessarily monotone in β.

Non-monotonicity of $\bar{w}_B$

In the system $S^{(A,(B,C))}(\beta, 1)$ we would expect that if β increases, i.e., the bandwidth of class A increases, that this would be at the expense of both classes B and C. However, it is possible for $\bar{w}_B$ to decrease as β increases; although it is counterintuitive. This possibility is, for instance, what we see in slot 4 of the example in Figure 3.6.
In Figure 3.7, we show another complete counterexample to demonstrate the possibility of $\bar{w}_B$ decreasing as β increases. Suppose the events in these 8 slots are repeated ad infinitum. Then by increasing β to $\beta + \Delta\beta$ (in the counterexample), we get that $\bar{w}_A$ decreases from 0.25 to 0.125 and $\bar{w}_C$ increases from 0.125 to 0.75. Counterintuitively $\bar{w}_B$ also decreases from 1.75 to 1.25.
To end up in this scenario it is necessary for class A to be empty and B,C non-empty in the $\beta + \Delta\beta$-system whilst in the β-system all are non-empty. At that particular slot the decision variable has to draw to serve class A. For a full survey of the possibilities we refer to the formal proof. It is however clear that the situation wherein $\bar{w}_B$ decreases as β increases in $S^{(A,(B,C))}(\beta, 1)$ over the full range of possible scenarios is unlikely for *regular* systems. We will use this in our construction of an optimization algorithm in Section 3.3.

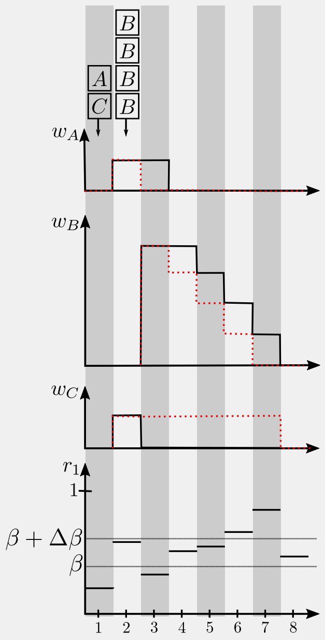


Figure 3.7: Counterexample decreasing $\bar{w}_B$ for increasing β.

The example and explanation of Figure 3.6 demonstrate the structure and most important cases of the proof of Lemma 3.1. We now state the formal proof.

Proof. Assume two coupled systems $S^{(A,(B,C))}(\beta, 1)$ and $S^{(A,(B,C))}(\beta+$

$\Delta\beta, 1)$ $(0 \leq \beta < \beta+\Delta\beta \leq 1)$, both systems have the same number of arrivals in each slot and use the same decision variables for scheduling. We now study the system starting from a certain slot, whereby this slot is such that the unfinished work of corresponding classes in both systems is equal. This can be done without loss of generality, as it is the case in the first slot of a regeneration cycle.

We study the sample paths of both systems from slot to slot. We keep track of the difference vector in slot k defined as: $\boldsymbol{\Delta w}_k = \boldsymbol{w}_k^{(A,(B,C))}(\beta+\Delta\beta,1)-\boldsymbol{w}_k^{(A,(B,C))}(\beta,1)$. In Tables 3.1-3.5, we describe starting from a certain difference vector $\boldsymbol{\Delta w}_k$ (indicated at the top of the table) the possibilities for the difference vector in the next slot (i.e.: $\boldsymbol{\Delta w}_{k+1}$) and under which conditions these transitions occur. We will explain the parts marked in grey in extenso to clarify the tables further.

When the difference vector at the start of the slot equals $(0,0,0)$ both systems are synchronized and they will stay synchronized when $r_k \notin [\beta, \beta+\Delta\beta[$. In Table 3.1, we only list the cases where $r_k \in [\beta, \beta+\Delta\beta[$. As an example, we explain the case marked in grey. When class B is empty and classes A and C backlogged in both systems ($\boldsymbol{\Delta w}_k = (0,0,0)$), $S^{(A,(B,C))}(\beta,1)$ serves class C and $S^{(A,(B,C))}(\beta+\Delta\beta,1)$ serves class A. As a consequence $\boldsymbol{\Delta w}_{k+1} = (-1,0,1)$.

The difference vector of $(-1,0,1)$ is of the form $(-m,0,m)$ with $(m>0)$, transition possibilities for this case are listed in Table 3.2. We focus on the case in grey for illustration purposes. When $w_{A,k}^{(A,(B,C))}(\beta+\Delta\beta,1) = 0, w_{B,k}^{(A,(B,C))}(\beta+\Delta\beta,1) > 0, w_{C,k}^{(A,(B,C))}(\beta+\Delta\beta,1) \geq m$, we know (as the difference vector equals $(-m,0,m)$) $w_{A,k}^{(A,(B,C))}(\beta+\Delta\beta,1) = m, w_{B,k}^{(A,(B,C))}(\beta+\Delta\beta,1) > 0, w_{C,k}^{(A,(B,C))}(\beta+\Delta\beta,1) \geq 0$. As a consequence, $S^{(A,(B,C))}(\beta+\Delta\beta,1)$ always serves class B, but $S^{(A,(B,C))}(\beta,1)$ serves class A when $r_k < \beta$ and class B otherwise. Consequently, $\boldsymbol{\Delta w}_{k+1} = (-m+1,-1,m)$ when $r_k < \beta$ and $\boldsymbol{\Delta w}_{k+1} = \boldsymbol{\Delta w}_k$ otherwise. When $w_{A,k}^{(A,(B,C))}(\beta+\Delta\beta,1), w_{B,k}^{(A,(B,C))}(\beta+\Delta\beta,1) > 0, w_{C,k}^{(A,(B,C))}(\beta+\Delta\beta,1) \geq m$, we know that $w_{A,k}^{(A,(B,C))}(\beta+\Delta\beta,1) > m, w_{B,k}^{(A,(B,C))}(\beta+\Delta\beta,1) > 0, w_{C,k}^{(A,(B,C))}(\beta+\Delta\beta,1) \geq 0$. As none of the queues is empty in both systems they serve the same class when $r_k \notin [\beta, \beta+\Delta\beta[$. However, when $\beta \leq r_k < \beta+\Delta\beta$, $S^{(A,(B,C))}(\beta,1)$ serves class B and $S^{(A,(B,C))}(\beta+\Delta\beta,1)$ serves class A. Consequently, $\boldsymbol{\Delta w}_{k+1} = (-m-1,1,m)$ when $\beta \leq r_k < \beta+\Delta\beta$ and $\boldsymbol{\Delta w}_{k+1} = \boldsymbol{\Delta w}_k$ otherwise.

Analogously, Tables 3.3, 3.4 and 3.5 are constructed summarizing all possible starting scenarios of $\boldsymbol{\Delta w}_k$. We observe that the first

Table 3.1: Exhaustively listing all possible difference vector transitions in slot k starting from difference vector $(0,0,0)$ given that $\beta \le r_k < \beta + \Delta\beta$

$\boldsymbol{\Delta w_k = (0,0,0)}$ and $\beta \le r_k < \beta + \Delta\beta$

$w_{\cdot,k}^{(A,(B,C))}(\beta+\Delta\beta,1)$			$\boldsymbol{\Delta w}_{k+1}$
A	B	C	
0	≥ 0	≥ 0	$(0,0,0)$
≥ 0	0	0	$(0,0,0)$
> 0	0	> 0	$(-1,0,1)$
> 0	> 0	≥ 0	$(-1,1,0)$

Table 3.2: Exhaustively listing all possible difference vector transitions in slot k starting from difference vector $(-m,0,m)$ with $(m>0)$

$\boldsymbol{\Delta w_k = (-m,0,m)}; m > 0$

$w_{\cdot,k}^{(A,(B,C))}(\beta+\Delta\beta,1)$				$\boldsymbol{\Delta w}_{k+1}$
A	B	C	r_k	
0	0	m		$(-m+1,0,m-1)$
0	0	$> m$	$r_k < \beta$	$(-m+1,0,m-1)$
			else	$(-m,0,m)$
0	> 0	$\ge m$	$r_k < \beta$	$(-m+1,-1,m)$
			else	$(-m,0,m)$
> 0	0	m	$r_k \ge \beta+\Delta\beta$	$(-m+1,0,m-1)$
			else	$(-m,0,m)$
> 0	0	$> m$	$\beta \le r_k < \beta+\Delta\beta$	$(-m-1,0,m+1)$
			else	$(-m,0,m)$
> 0	> 0	$\ge m$	$\beta \le r_k < \beta+\Delta\beta$	$(-m-1,1,m)$
			else	$(-m,0,m)$

Table 3.3: Exhaustively listing all possible difference vector transitions in slot k starting from difference vector $(0,-m,m)$ with $(m>0)$

$\boldsymbol{\Delta w_k = (0,-m,m)}; m > 0$

$w_{\cdot,k}^{(A,(B,C))}(\beta+\Delta\beta,1)$				$\boldsymbol{\Delta w}_{k+1}$
A	B	C	r_k	
0	0	$\ge m$		$(0,-m+1,m-1)$
0	> 0	$\ge m$		$(0,-m,m)$
> 0	0	$\ge m$	$r_k \ge \beta+\Delta\beta$	$(-1,-m+1,m)$
			$r_k < \beta$	$(0,-m,m)$
			$r_k \ge \beta+\Delta\beta$	$(0,-m+1,m-1)$
> 0	> 0	$\ge m$	$\beta \le r_k < \beta+\Delta\beta$	$(-1,-m+1,m)$
			else	$(0,-m,m)$

Table 3.4: Exhaustively listing all possible difference vector transitions in slot k starting from difference vector $(-m-n, m, n)$ with $(m > 0, n \geq 0)$

$\boldsymbol{\Delta w_k = (-m-n, m, n)}; m > 0, n \geq 0$

$w_{\cdot,k}^{(A,(B,C))}(\beta+\Delta\beta,1)$ A	B	C	r_k	$\boldsymbol{\Delta w}_{k+1}$
0	m	n		$(-m-n+1, m-1, n)$
0	m	$> n$	$r_k < \beta$	$(-m-n+1, m-1, n)$
			else	$(-m-n, m-1, n+1)$
0	$> m$	$\geq n$	$r_k < \beta$	$(-m-n+1, m-1, n)$
			else	$(-m-n, m, n)$
> 0	m	n	$r_k \geq \beta+\Delta\beta$	$(-m-n+1, m-1, n)$
			else	$(-m-n, m, n)$
> 0	m	$> n$	$\beta \leq r_k < \beta+\Delta\beta$	$(-m-n-1, m, n+1)$
			$r_k < \beta$	$(-m-n, m, n)$
			$r_k \geq \beta+\Delta\beta$	$(-m-n, m-1, n+1)$
> 0	$> m$	$\geq n$	$\beta \leq r_k < \beta+\Delta\beta$	$(-m-n-1, m+1, n)$
			else	$(-m-n, m, n)$

Table 3.5: Exhaustively listing all possible difference vector transitions in slot k starting from difference vector $(-m, -n, m+n)$ with $(m, n > 0)$

$\boldsymbol{\Delta w_k = (-m, -n, m+n)}; m, n > 0$

$w_{\cdot,k}^{(A,(B,C))}(\beta+\Delta\beta,1)$ A	B	C	r_k	$\boldsymbol{\Delta w}_{k+1}$
0	0	$\geq m+n$	$r_k < \beta$	$(-m+1, -n, m+n-1)$
			else	$(-m, -n+1, m+n-1)$
0	> 0	$\geq m+n$	$r_k < \beta$	$(-m+1, -n-1, m+n)$
			else	$(-m, -n, m+n)$
> 0	0	$\geq m+n$	$r_k \geq \beta+\Delta\beta$	$(-m-1, -n+1, m+n)$
			$r_k < \beta$	$(-m, -n, m+n)$
			$r_k \geq \beta+\Delta\beta$	$(-m, -n+1, m+n-1)$
> 0	> 0	$\geq m+n$	$\beta \leq r_k < \beta+\Delta\beta$	$(-m-1, -n+1, m+n)$
			else	$(-m, -n, m+n)$

element in the possible difference vectors is always negative and the third element is always positive. The second element can be positive or negative, which makes further conclusions for w_B impossible. As such, we conclude that for every slot k in the sample path for the coupled systems:

$$\begin{aligned} w_{A,k}^{(A,(B,C))}(\beta,1) &\geq w_{A,k}^{(A,(B,C))}(\beta+\Delta\beta,1), \\ w_{C,k}^{(A,(B,C))}(\beta,1) &\leq w_{C,k}^{(A,(B,C))}(\beta+\Delta\beta,1). \end{aligned} \tag{3.2}$$

And thus, their means satisfy:

$$\begin{aligned} \bar{w}_{A}^{(A,(B,C))}(\beta,1) &\geq \bar{w}_{A}^{(A,(B,C))}(\beta+\Delta\beta,1), \\ \bar{w}_{C}^{(A,(B,C))}(\beta,1) &\leq \bar{w}_{C}^{(A,(B,C))}(\beta+\Delta\beta,1). \end{aligned} \tag{3.3}$$

□

3.2.2 Comparison of the interior boundary for two H-GPS systems

Lemma 3.2. *For* (A,B,C) *an arbitrary permutation of* $(1,2,3)$, $\boldsymbol{\bar{w}^{(A,(B,C))}(1,0)} = \boldsymbol{\bar{w}^{(B,(A,C))}(0,1)}$. *Furthermore we have for* $\beta \in]0,1]$ *that:*

(1) $\bar{w}_A^{(A,(B,C))}(1-\beta,0) \geq \bar{w}_A^{(B,(A,C))}(\beta,1)$

(2) $\bar{w}_B^{(A,(B,C))}(1-\beta,0) \geq \bar{w}_B^{(B,(A,C))}(\beta,1)$

(3) $\bar{w}_C^{(A,(B,C))}(1-\beta,0) \leq \bar{w}_C^{(B,(A,C))}(\beta,1)$

In the proof of this lemma, the difference vector $\boldsymbol{w}^{(B,(A,C))}(\beta,1) - \boldsymbol{w}^{(A,(B,C))}(1-\beta,0)$ is analyzed sample-path wise. Studying this difference vector is equivalent to comparing the sample paths of both $\boldsymbol{w}^{(B,(A,C))}(\beta,1)$ and $\boldsymbol{w}^{(A,(B,C))}(1-\beta,0)$. The proof couples two systems: $S^{(A,(B,C))}(1-\beta,0)$ and $S^{(B,(A,C))}(\beta,1)$. To couple the decision variables $S^{(B,(A,C))}(\beta,1)$ uses $r_{1,k}$ and $S^{(A,(B,C))}(1-\beta,0)$ uses $1-r_{1,k}$. That way, when $r_{1,k} \geq \beta$, class A gets served in both $S^{(B,(A,C))}(\beta,1)$ and $S^{(A,(B,C))}(1-\beta,0)$ $(1-r_{1,k} \leq 1-\beta)$. When no A-customers are present both systems serve class C. As such, $S^{(B,(A,C))}(\beta,1)$ and $S^{(A,(B,C))}(1-\beta,0)$ only make different decisions when $r_{1,k} < \beta$. With $\beta = 0$, both systems correspond to the priority ordering $(A > C > B)$ policy, consequently $\boldsymbol{\bar{w}^{(A,(B,C))}(1,0)} = \boldsymbol{\bar{w}^{(B,(A,C))}(0,1)}$.

In Figure 3.8, we present a samplepath containing all (blueprints of) possible difference vectors as an example. In the figure, we start

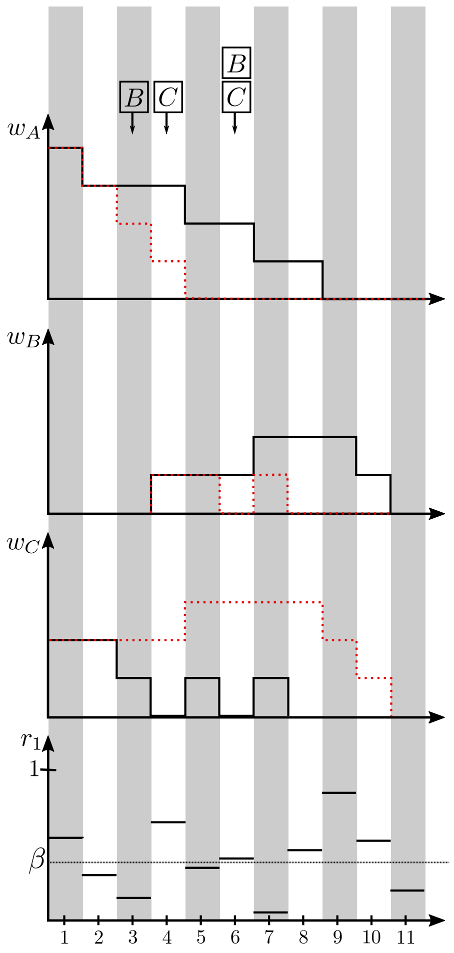


Figure 3.8: Illustration of Lemma 3.2 $S^{(B,(A,C))}(\beta,1)$ (dotted) $S^{(A,(B,C))}(1-\beta,0)$ (solid)

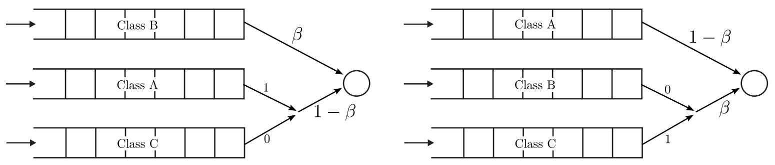


Figure 3.9: Systems $S^{(B,(A,C))}(\beta, 1)$ (left) and $S^{(A,(B,C))}(1-\beta, 0)$ (right) as considered in the proof of Lemma 3.2.

from a situation where the sample paths of both systems $S^{(B,(A,C))}$ $(\beta, 1)$ and $S^{(A,(B,C))}(1-\beta, 0)$ are aligned, and the difference vector is $(0, 0, 0)$. In the second slot $r_1 < \beta$ and $S^{(A,(B,C))}(1-\beta, 0)$ serves class C, while $S^{(B,(A,C))}(\beta, 1)$ serves class A (class B is empty). The result is a difference vector of $(-1, 0, 1)$. The same happens in slot 3, adding up to a difference vector of $(-2, 0, 2)$. These differences stay constant up to the first slot where $r_1 < \beta$ and class B is non-empty. At this point, $S^{(A,(B,C))}(1-\beta, 0)$ serves class C, while $S^{(B,(A,C))}(\beta, 1)$ serves class B, resulting in a difference vector of $(-2, -1, 3)$. In the next slot, $S^{(A,(B,C))}(1-\beta, 0)$ serves class A, while $S^{(B,(A,C))}(\beta, 1)$ serves class C (class A is empty in this slot in this system). The difference vector now (in slot 7) is $(-1, -1, 2)$ and we get the same situation as in slot 5, amounting to a difference vector of $(-1, -2, 3)$. From this point on $r_1 \geq \beta$ and both systems empty, aligning them both when they are empty. Looking at the possible difference vectors, the inequalities from Lemma 3.2 are clear.

Similar as for Lemma 3.1, the example we just discussed demonstrates the structure of the proof of Lemma 3.2. For completeness, we write down the formal proof below.

Proof. Assume two coupled systems $S^{(B,(A,C))}(\beta, 1)$ and $S^{(A,(B,C))}$ $(1-\beta, 0)$, where both systems have the same arrivals in each slot. To couple the decision variables in slot l, $S^{(B,(A,C))}(\beta, 1)$ uses r_l and $S^{(A,(B,C))}(1-\beta, 0)$ uses $1-r_l$. We now study the system starting from a certain slot, whereby this slot is such that the unfinished work of corresponding classes in both systems is equal. This can be done without loss of generality, as this could very well be the first slot of a regeneration cycle.

We study the sample path of both systems from slot to slot, whereby the current slot is denoted as k. We keep track of the difference vector in the current slot k defined as: $\boldsymbol{\Delta w}_k = \boldsymbol{w}_k^{(B,(A,C))}(\beta, 1) - \boldsymbol{w}_k^{(A,(B,C))}(1-\beta, 0)$. In Tables 3.6-3.9, we describe starting from a certain difference vector $\boldsymbol{\Delta w}_k$ (indicated at the top of the table) the possibilities for the difference vector in the next slot (i.e.: $\boldsymbol{\Delta w}_{k+1}$) and under which conditions these transitions occur. We will explain the parts marked in grey in extenso to clarify the tables further. Furthermore, for brevity, we drop the arguments $(1-\beta, 0)$ for the $(A,(B,C))$ system and $(\beta, 1)$ for the $(B,(A,C))$ system in the remainder of the proof.

As a first example, we start from synchronized sample paths so the difference vector is $(0,0,0)$, subsequently, the possibilities in that slot are listed in Table 3.6. We study the case in grey. The first three columns indicate that $w_{A,k}^{(B,(A,C))}, w_{B,k}^{(B,(A,C))}, w_{C,k}^{(B,(A,C))} > 0$ as the difference vector in the current slot is $(0,0,0)$ this means that also $w_{A,k}^{(A,(B,C))}, w_{B,k}^{(A,(B,C))}, w_{C,k}^{(A,(B,C))} > 0$. With probability $1-\beta$, both systems serve class A, keeping the systems aligned and $\boldsymbol{\Delta w}_k = (0,0,0)$. With probability β, $(A,(B,C))$ serves class C, while $(B,(A,C))$ serves class B. This results in a difference vector $(0,-1,1)$ for the next slot. From the complete table we see that from difference vector $(0,0,0)$ the systems can either stay aligned in the next slot or evolve to difference vectors $(0,-1,1)$ or $(-1,0,1)$.

Vector $(0,-1,1)$ is of the form $(0,-m,m)$ (with $m>0$), and we indicate the possibilities from this difference vector onwards in Table 3.7. As a second example, we focus on the case where $w_{A,k}^{(B,(A,C))} > 0, w_{B,k}^{(B,(A,C))} = 0, w_{C,k}^{(B,(A,C))} = m$ (indicated in grey) as the current difference vector is $(0,-m,m)$, $w_{A,k}^{(A,(B,C))} > 0, w_{B,k}^{(A,(B,C))} = m$, $w_{C,k}^{(A,(B,C))} = 0$, see Table 3.7. With probability β, $(A,(B,C))$ serves class B and $(B,(A,C))$ serves class A, in the next slot the difference vector thus equals $(-1,-m+1,m)$. With probability $1-\beta$, both systems serve class A and the difference vector stays as is, i.e.: $(0,-m,m)$.

Analogous to these two examples the complete Tables 3.6 - 3.9 can be built. Comparing the sample paths of both coupled systems there are only 4 possible (blueprints of) difference vectors for every slot : $(0,0,0), (-m,0,m), (0,-m,m)$ and $(-m,-n,m+n)$ with $m,n>0$. As such, we conclude that for every slot k in the sample path for the coupled systems:

$$\begin{aligned} w_{A,k}^{(A,(B,C))} &\geq w_{A,k}^{(B,(A,C))}, \\ w_{B,k}^{(A,(B,C))} &\geq w_{B,k}^{(B,(A,C))}, \\ w_{C,k}^{(A,(B,C))} &\leq w_{C,k}^{(B,(A,C))}. \end{aligned} \tag{3.4}$$

Table 3.6: Exhaustively listing all difference vector transitions starting from difference vectors $(0, 0, 0)$

$\boldsymbol{\Delta w_k = (0, 0, 0)}$

$w_{\cdot,k}^{(B,(A,C))}$ A	B	C	w.p.	$\boldsymbol{\Delta w}_{k+1}$
>0	0	0	1	$(0, 0, 0)$
0	>0	0	1	$(0, 0, 0)$
0	0	>0	1	$(0, 0, 0)$
0	>0	>0	β	$(0, -1, 1)$
			$1-\beta$	$(0, 0, 0)$
>0	0	>0	β	$(-1, 0, 1)$
			$1-\beta$	$(0, 0, 0)$
>0	>0	0	1	$(0, 0, 0)$
>0	>0	>0	β	$(0, -1, 1)$
			$1-\beta$	$(0, 0, 0)$

Table 3.7: Exhaustively listing all difference vector transitions starting from difference vectors $(0, -m, m)$ $(m > 0)$

$\boldsymbol{\Delta w_k = (0, -m, m)}$

$w_{\cdot,k}^{(B,(A,C))}$ A	B	C	w.p.	$\boldsymbol{\Delta w}_{k+1}$
0	0	m	1	$(0, -m+1, m-1)$
0	0	$>m$	1	$(0, -m, m)$
0	>0	m	β	$(0, -m, m)$
			$1-\beta$	$(0, -m+1, m-1)$
0	>0	$>m$	β	$(0, -m-1, m+1)$
			$1-\beta$	$(0, -m, m)$
>0	0	m	β	$(-1, -m+1, m)$
			$1-\beta$	$(0, -m, m)$
>0	0	$>m$	β	$(-1, -m, m+1)$
			$1-\beta$	$(0, -m, m)$
>0	>0	m	1	$(0, -m, m)$
>0	>0	$>m$	β	$(0, -m-1, m+1)$
			$1-\beta$	$(0, -m, m)$

Table 3.8: Exhaustively listing all difference vector transitions starting from difference vectors $(-m, -n, m+n)$ $(m, n > 0)$

$\boldsymbol{\Delta w_k = (-m, -n, m+n)}$

$w_{\cdot,k}^{(B,(A,C))}$				
A	B	C	w.p.	$\boldsymbol{\Delta w}_{k+1}$
0	0	$m+n$	β	$(-m, -n+1, m+n-1)$
			$1-\beta$	$(-m+1, -n, m+n-1)$
0	0	$> m+n$	β	$(-m, -n, m+n)$
			$1-\beta$	$(-m+1, -n, m+n-1)$
0	> 0	$m+n$	β	$(-m, -n, m+n)$
			$1-\beta$	$(-m+1, -n, m+n-1)$
0	> 0	$> m+n$	β	$(-m, -n-1, m+n+1)$
			$1-\beta$	$(-m+1, -n, m+n-1)$
> 0	0	$m+n$	β	$(-m-1, -n+1, m+n)$
			$1-\beta$	$(-m, -n, m+n)$
> 0	0	$> m+n$	β	$(-m-1, -n, m+n+1)$
			$1-\beta$	$(-m, -n, m+n)$
> 0	> 0	$m+n$	1	$(-m, -n, m+n)$
> 0	> 0	$> m+n$	β	$(-m, -n-1, m+n+1)$
			$1-\beta$	$(-m, -n, m+n)$

Table 3.9: Exhaustively listing all difference vector transitions starting from difference vectors $(-m, 0, m)$ $(m > 0)$

$\boldsymbol{\Delta w_k = (-m, 0, m)}$

$w_{\cdot,k}^{(B,(A,C))}$				
A	B	C	w.p.	$\boldsymbol{\Delta w}_{k+1}$
0	0	m	1	$(-m+1, 0, m-1)$
0	0	$> m$	β	$(-m, 0, m)$
			$1-\beta$	$(-m+1, 0, m-1)$
0	> 0	m	β	$(-m, 0, m)$
			$1-\beta$	$(-m+1, 0, m-1)$
0	> 0	$> m$	β	$(-m, -1, m+1)$
			$1-\beta$	$(-m+1, 0, m-1)$
> 0	0	m	1	$(-m, 0, m)$
> 0	0	$> m$	β	$(-m-1, 0, m+1)$
			$1-\beta$	$(-m, 0, m)$
> 0	> 0	m	1	$(-m, 0, m)$
> 0	> 0	$> m$	β	$(-m, -1, m+1)$
			$1-\beta$	$(-m, 0, m)$

And thus, their means satisfy:

$$\begin{aligned}\bar{w}_A^{(A,(B,C))}(1-\beta,0) &\geq \bar{w}_A^{(B,(A,C))}(\beta,1),\\ \bar{w}_B^{(A,(B,C))}(1-\beta,0) &\geq \bar{w}_B^{(B,(A,C))}(\beta,1),\\ \bar{w}_C^{(A,(B,C))}(1-\beta,0) &\leq \bar{w}_C^{(B,(A,C))}(\beta,1). \qquad (3.5)\end{aligned}$$

□

3.2.3 Proof of Theorem 3.1

With the aid of Lemmas 3.1 and 3.2, we can now proceed to the proof of Theorem 3.1.

Proof of Theorem 3.1. Take the performance vector $\bar{\boldsymbol{w}}^* = (\bar{w}_1^*, \bar{w}_2^*, \bar{w}_3^*)$ in the polytope Ω. Then $\exists b_1, b_2, b_3 : \bar{w}_1^* = \bar{w}_1^{(1,(2,3))}(b_1, \cdot), \bar{w}_2^* = \bar{w}_2^{(2,(1,3))}(b_2, \cdot), \bar{w}_3^* = \bar{w}_3^{(3,(1,2))}(b_3, \cdot)$.

There are three possibilities with respect to $\bar{\boldsymbol{w}}^*$ and H-GPS. Take one of the configurations, e.g. $(1,(2,3))$:

- $\bar{w}_2^{(1,(2,3))}(b_1,1) \leq \bar{w}_2^* \leq \bar{w}_2^{(1,(2,3))}(b_1,0)$
  In this case $\exists b_1' : \bar{w}_2^* = \bar{w}_2^{(1,(2,3))}(b_1, b_1')$ and because of the work-conserving property $\bar{w}_3^* = \bar{w}_3^{(1,(2,3))}(b_1, b_1')$. As such $\bar{\boldsymbol{w}}^*$ is feasible with system $(1,(2,3))$ and parameter combination (b_1, b_1').

- $\bar{w}_2^* < \bar{w}_2^{(1,(2,3))}(b_1,1)$.
  From the work-conserving property, it follows $\bar{w}_3^* > \bar{w}_3^{(1,(2,3))}(b_1,1)$. Since $\bar{w}_2^* = \bar{w}_2^{(2,(1,3))}(b_2,0)$, we find that $\bar{w}_2^{(1,(2,3))}(b_1,1) \leq \bar{w}_2^{(2,(1,3))}(1-b_1,0)$, by means of Lemma 3.2. Then, Lemma 3.1 yields $b_2 > 1-b_1$. Analogously $b_3 < 1-b_1$ as $\bar{w}_3^* = \bar{w}_3^{(3,(1,2))}(b_3,1) > \bar{w}_3^{(1,(2,3))}(b_1,1) \geq \bar{w}_3^{(3,(1,2))}(1-b_1,1)$. To summarize: $\bar{\boldsymbol{w}}^*$ is not feasible with system $(1,(2,3))$ in this case and $b_3 < 1-b_1 < b_2$.

- $\bar{w}_2^* > \bar{w}_2^{(1,(2,3))}(b_1,0)$ and thus $\bar{w}_3^* < \bar{w}_3^{(1,(2,3))}(b_1,0)$
  Analogously as in the previous case we get that $\bar{\boldsymbol{w}}^*$ is not feasible with system $(1,(2,3))$ and that $b_3 > 1-b_1 > b_2$.

For H-GPS $(2,(1,3))$ we have:

- $\bar{w}_1^{(2,(1,3))}(b_2,1) \leq \bar{w}_1^* \leq \bar{w}_1^{(2,(1,3))}(b_2,0)$
  In this case $\exists b_2' : \bar{w}_1^* = \bar{w}_1^{(2,(1,3))}(b_2, b_2')$ and because of the

work-conserving property $\bar{w}_3^* = \bar{w}_3^{(2,(1,3))}(b_2, b_2')$. As such $\bar{\boldsymbol{w}}^*$ is feasible with system $(2,(1,3))$ and parameter combination (b_2, b_2').

- $\bar{w}_1^* < \bar{w}_1^{(2,(1,3))}(b_2, 1)$ and $\bar{w}_3^* > \bar{w}_3^{(2,(1,3))}(b_2, 1)$
  Analogous to $(1,(2,3))$, we have that $b_1 > 1-b_2$ and $1-b_2 > b_3$. Which summarizes to: $b_3 < 1 - b_2 < b_1$ and that $\bar{\boldsymbol{w}}^*$ cannot be achieved with system $(2,(1,3))$.

- $\bar{w}_1^* > \bar{w}_1^{(2,(1,3))}(b_2, 0)$ and $\bar{w}_3^* < \bar{w}_3^{(2,(1,3))}(b_2, 0)$
  Analogously, we get that $\bar{\boldsymbol{w}}^*$ cannot be reached with system $(2,(1,3))$ and that $b_1 < 1 - b_2 < b_3$.

Lastly for H-GPS $(3,(1,2))$ we have:

- $\bar{w}_1^{(3,(1,2))}(b_3, 1) \leq \bar{w}_1^* \leq \bar{w}_1^{(3,(1,2))}(b_3, 0)$
  In this case $\exists b_3'$ for which $\bar{\boldsymbol{w}}^*$ is feasible with system $(3,(1,2))$ and parameter combination (b_3, b_3').

- $\bar{w}_1^* < \bar{w}_1^{(3,(1,2))}(b_3, 1)$ and $\bar{w}_2^* > \bar{w}_2^{(3,(1,2))}(b_3, 1)$
  Consequently, $\bar{\boldsymbol{w}}^*$ is not feasible with system $(3,(1,2))$ and $b_2 < 1 - b_3 < b_1$.

- $\bar{w}_1^* > \bar{w}_1^{(3,(1,2))}(b_3, 0)$ and $\bar{w}_2^* < \bar{w}_2^{(3,(1,2))}(b_3, 0)$
  As a consequence, $\bar{\boldsymbol{w}}^*$ cannot be reached with system $(3,(1,2))$ and $b_1 < 1 - b_3 < b_2$.

We now combine all possibilities; this is shown in the tree structure in Figure 3.10. It is easily seen that the combinations where $\bar{\boldsymbol{w}}^*$ is infeasible with all three H-GPS systems lead to a contradiction. As an example, we focus on the topmost path with three edges in the tree from the figure. In this case $\bar{\boldsymbol{w}}^*$ is infeasible with all three H-GPS systems but we also obtain the inequalities from (3.6). Clearly the first and last equations are contradictory, proving this combination is impossible. As all paths whereby $\bar{\boldsymbol{w}}^*$ is infeasible lead to contradictions, only paths where $\bar{\boldsymbol{w}}^*$ is feasible by one of the H-GPS systems are valid, concluding the proof.

$$\begin{cases} b_2 < 1 - b_1 < b_3 \\ b_1 < 1 - b_2 < b_3 \\ b_1 < 1 - b_3 < b_2 \end{cases} \tag{3.6}$$

□

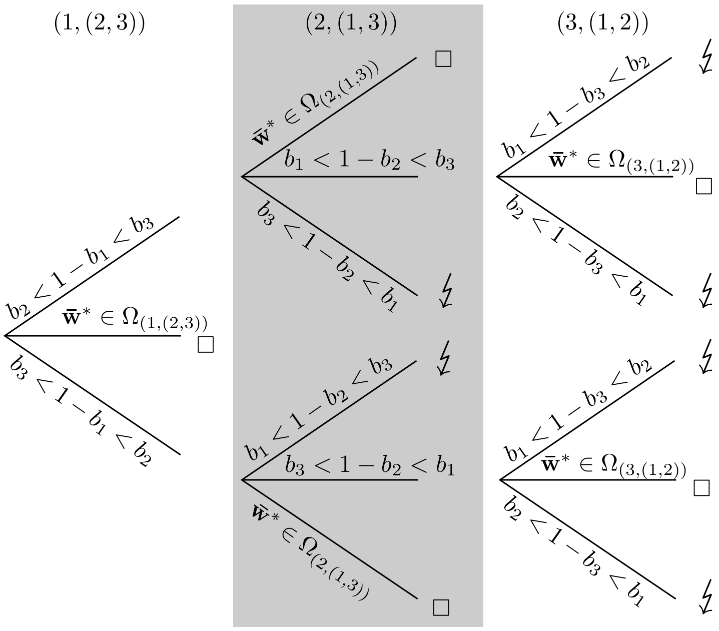


Figure 3.10: Tree showing possibilities for the proof of Theorem 3.1

An alternative way of looking at the proof of Theorem 3.1

If we look again at Figure 3.5, we establish that every H-GPS feasible region is bounded by 2 straight lines of the polytope Ω and 2 curves in the interior of Ω (these curves where obtained via simulation). For the straight lines of the achievable region of $S^{(A,(B,C))}(\beta_1, \beta_2)$, we have that β_1 equals either 0 or 1 while β_2 traverses from 0 to 1. Clearly this is a boundary of Ω as we keep highest (lowest) priority for class A while investigating all work-conserving possibilities for $(\bar{w}_B, \bar{w}_C)$. For the curves bounding the achievable region of $S^{(A,(B,C))}(\beta_1, \beta_2)$, we have that β_1 traverses from 0 to 1 while β_2 equals either 0 or 1. For instance, $S^{(A,(B,C))}(0 \to 1, 1)$ results in a curve $(B > C > A) \to (A > B > C)$ which indeed travels through the interior of Ω. The fact that these lines are curves shows that β_1 indeed impacts all of the queues in the system.

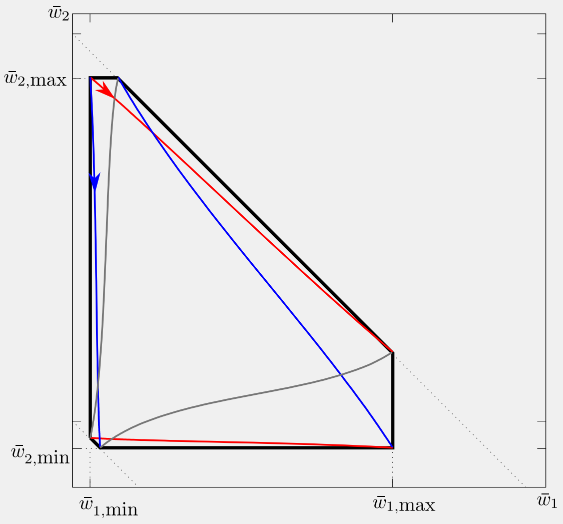


Figure 3.11: Interior boundaries to Ω of H-GPS systems.

In Figure 3.11, we have drawn all these interior curves that bound one of the achievable regions of a H-GPS system. We see that each vertex of the polytope Ω is an endpoint to two curves of different H-GPS systems. If we can prove that none of the curves, that have an endpoint in common, intersect creating a *gap*, the whole polytope Ω is necessarily achieved by at least one of the H-GPS systems (try it!). This can easily be proven using Lemmas 3.1 and 3.2. Lemma 3.2, in fact, compares two of these curves (this also explains the title of Section 3.2.2).

For instance, if we apply Lemma 3.2 to $(A, B, C) = (1, 2, 3)$, then $\bar{\boldsymbol{w}}^{(1,(2,3))}(1, 0) = \bar{\boldsymbol{w}}^{(2,(1,3))}(0, 1) = \bar{\boldsymbol{w}}^{(1>3>2)}$. The curves $S^{(1,(2,3))}(1 - \beta, 0)$ and $S^{(2,(1,3))}(\beta, 1)$ are indicated by and flow in the direction of the arrows in Figure 3.11, if β traverses from 0 to 1. If these lines were to intersect, then for some b_1 and b_2 we have that

$$\bar{w}_1^{(1,(2,3))}(1 - b_1, 0) = \bar{w}_1^{(2,(1,3))}(b_2, 1), \tag{3.7}$$

from Lemma 3.2 we get that

$$\bar{w}_1^{(1,(2,3))}(1-b_2,0) \geq \bar{w}_1^{(2,(1,3))}(b_2,1). \tag{3.8}$$

Combining (3.7) and (3.8) and applying Lemma 3.1, we find that $b_2 \geq b_1$. Subsequently, we find

$$\bar{w}_2^{(1,(2,3))}(1-b_1,0) \geq \bar{w}_2^{(2,(1,3))}(b_1,1) \geq \bar{w}_2^{(2,(1,3))}(b_2,1),$$

as $b_2 \geq b_1$, using Lemma 3.2 for the right-hand inequality and Lemma 3.1 for the left-hand inequality . Summarized, we find that for any b_1 and b_2 for which $\bar{w}_1^{(1,(2,3))}(1-b_1,0) = \bar{w}_1^{(2,(1,3))}(b_2,1)$, $\bar{w}_2^{(1,(2,3))}(1-b_1,0) \geq \bar{w}_2^{(2,(1,3))}(b_2,1)$, as a result one curve is always above (or on) the other and it is impossible to create a *gap* that possibly creates an inachievable zone by H-GPS in Ω.

3.3 Optimization

Given a certain cost function $f(\bar{\boldsymbol{w}})$, we want to find the H-GPS system and corresponding parameters β_1 and β_2 that lead to a minimum cost. This cost function combined with the knowledge of the total performance region of any H-GPS system (feasible values for $\bar{\boldsymbol{w}}$), which was the subject of the previous section, makes it possible to optimize the cost function in two steps. In the first step, we search the performance vector $\bar{\boldsymbol{w}}^*$ for which the cost function is minimal, i.e. $\bar{\boldsymbol{w}}^* = \operatorname{argmin}_{\bar{\boldsymbol{w}}\in\Omega} f(\bar{\boldsymbol{w}})$. This is an optimization problem that can be solved using standard techniques from linear or non-linear programming (depending on the form of the cost function) [16, 116, 132]. In a second step, we need to find a H-GPS system and the parameter combination (β_1, β_2) that achieves the optimal performance vector $\bar{\boldsymbol{w}}^*$, since we proved in the previous section that at least one of the three H-GPS systems can achieve $\bar{\boldsymbol{w}}^*$. This second step is the subject of this section.

We are left with two problems. Given a certain desired performance $\bar{\boldsymbol{w}}^*$, we answer (i) how to select a H-GPS system that achieves this and (ii) how to find the corresponding parameters β_1 and β_2 for that system. For (i), we start by identifying some regions of Ω that are certain to be achieved by a given H-GPS system. For instance, we are certain that H-GPS $(1,(2,3))$ achieves the region governed by the following inequalities (whereby $\bar{w}_1$ and $\bar{w}_2$ denote the axis variables.)

$$\bar{w}_1 \leq \bar{w}_{1,\max}, \tag{3.9}$$

$$\bar{w}_2 \leq \bar{w}_2^{(1,(2,3))}(0,0), \tag{3.10}$$

$$\bar{w}_1 + \bar{w}_2 \geq \bar{w}_T - \bar{w}_3^{(1,(2,3))}(0,1). \tag{3.11}$$

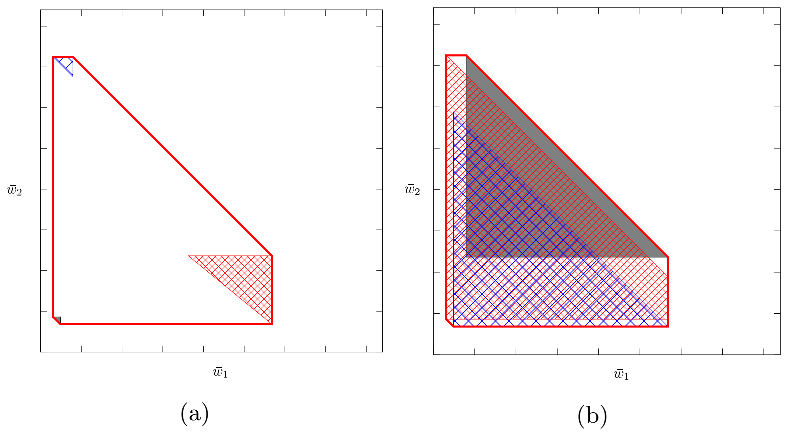


Figure 3.12: Regions certain (left) and likely (right) to be achievable by corresponding H-GPS systems.

(3.9) follows from Lemma 3.1 that dictates $\bar{w}_1^{(1,(2,3))}(\beta,0) \leq \bar{w}_1^{(1,(2,3))}(0,0) = \bar{w}_{1,\max}$. For each β the possible values for $\bar{w}_2$ are in $[\bar{w}_2^{(1,(2,3))}(\beta,1), \bar{w}_2^{(1,(2,3))}(\beta,0)]$. According to Lemma 3.1, $\bar{w}_2^{(1,(2,3))}(0,0) \leq \bar{w}_2^{(1,(2,3))}(\beta,0)$ and $\bar{w}_3^{(1,(2,3))}(0,1) \leq \bar{w}_3^{(1,(2,3))}(\beta,1)$. We use $\bar{w}_3^{(1,(2,3))}(\beta,1) = \bar{w}_T - \bar{w}_1^{(1,(2,3))}(\beta,1) - \bar{w}_2^{(1,(2,3))}(\beta,1)$ in this last inequality. We thus find that every value in $[\bar{w}_T - \bar{w}_2^{(1,(2,3))}(\beta,0) - \bar{w}_3^{(1,(2,3))}(0,1), \bar{w}_2^{(1,(2,3))}(0,0)]$. This gives (3.10) and (3.11).

The other two systems yield analogous achievable regions. We have drawn these regions in Figure 3.12a for an example. From extensive simulations, we have also identified regions for which it is very likely that they are achievable by a particular H-GPS system. These are depicted in Figure 3.12b. The existence of points in these regions that are not achievable by the corresponding H-GPS system, demand that there are also points for which $\bar{w}_B^{(A,(B,C))}(\beta,1) > \bar{w}_B^{(A,(B,C))}(\beta+\Delta\beta,1)$. As we know from Lemma 3.1 and the proof thereof this is possible. However, the proof provides insight that the probability of this occurring is in general smaller than the alternative. This makes the existence of a significant amount of points in these regions not achievable by the corresponding H-GPS system unlikely. It is easily proven that these 6 regions (3 certain and 3 likely) always span the whole polytope Ω.

With regard to issue (ii), we exploit the property of H-GPS (opposed to GPS) that the performance of the class on the first hierarchical level is independent of the decision parameter on the second level. For instance in $S^{(A,(B,C))}(\beta_1,\beta_2)$, $\bar{w}_A^{(A,(B,C))}$ is a function of β_1 only, while $\bar{w}_B$ and $\bar{w}_C$ are functions of both β_1 and β_2. As a result, the optimization can be done hierarchically. First, one optimizes for β_1 so that $\bar{w}_A^* = \bar{w}_A^{(A,(B,C))}(\beta_1,\cdot)$. In a second step, β_2 is optimized so that $\bar{w}_B^* = \bar{w}_B^{(A,(B,C))}(\beta_1,\beta_2)$.

We bundled all this into Algorithm 3.1. The first step of the algorithm is to sort the H-GPS systems to likeliness of achieving the required performance vector (recall that we proved that at least one achieves it). Therefore, we first assess if the target performance vector $\bar{\boldsymbol{w}}^*$ is in one of the *certain* regions. If it is, these are placed first in the list L. If the target performance is not in one of the certain regions, we look at the *likely* regions. For further ordering, for instance when the target performance is in two or more likely regions, we consider the *distance* to the extrema on the first level of the H-GPS system. For instance for H-GPS $(1,(2,3))$, we calculate $\min(\bar{w}_1^* - \bar{w}_{1,\text{min}}, \bar{w}_{1,\text{max}} - \bar{w}_1^*)$ and compare it to the other distances calculated. This is illustrated in Figure 3.13. For the optimization steps in the algorithm, one can for instance use binary search. In binary search, one simulates for $\beta = 0.5$ and identifies in which interval $\bar{\boldsymbol{w}}^*$ lies: either $[0; 0.5]$ or $[0.5; 1]$. If it is in the first interval one simulates for $\beta = 0.25$ and so on (see also the description of the algorithm in the note on page 61). As such the interval (and uncertainty) halves each step.

```
Define: L=[H-GPS-1,H-GPS-2,H-GPS-3]
/* Sort list to likeliness of achieving $\bar{\boldsymbol{w}}^*$                */
sort(L);
for H-GPS-j in L do
    /* Optimize $\beta_1$ on first hierarchical level achieving $\bar{w}_j$   */
    $\beta_1$ = optimize(H-GPS-j,$\bar{\boldsymbol{w}}^*$);
    /* Check if H-GPS-j achieves $\bar{\boldsymbol{w}}^*$ ($\beta_2 = 0$ and $\beta_2 = 1$)   */
    if achievable(H-GPS-j,$\bar{\boldsymbol{w}}^*$,$\beta_1$) then
        /* Optimize $\beta_2$ on second hierarchical level             */
        $\beta_2$ = optimize(H-GPS-j,$\bar{\boldsymbol{w}}^*$,$\beta_1$);
        return [H-GPS-j,$\beta_1$,$\beta_2$];
    else
        /* Remove H-GPS-j from list and try next system          */
        remove(H-GPS-j, L);
    end if
end for
```

Algorithm 3.1: Algorithm to find a H-GPS system and configuration (β_1,β_2) that achieves $\bar{\boldsymbol{w}}^* = (\bar{w}_1^*, \bar{w}_2^*, \bar{w}_3^*)$.

We tested this algorithm by running it 2160 times for different

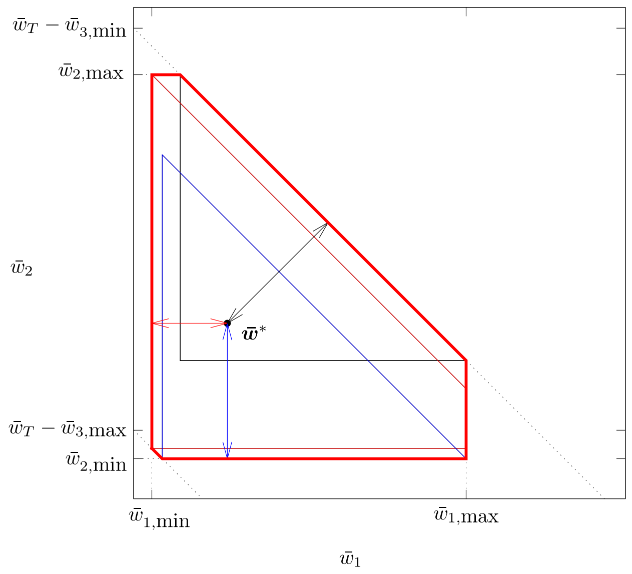


Figure 3.13: Illustration of the distance-criterium when $\bar{\boldsymbol{w}}^*$ is in multiple regions.

input configurations. As an arrival process we used a multinomial distribution for the number of arrivals as explained in Section 1.6, with $N = 16$, λ_T varying from 0.2 to 0.95 and α_i from 0.1 to 0.8 over the 2160 tests. For each test a $\bar{w}^*$ target was randomly chosen (uniformly) in Ω. Each optimization was done using binary search and with simulations over a sample path of 10^8 slots. The optimization was stopped when β_i was accurate up to 3 digits or $\bar{w}_i$ was within 0.1% of $\bar{w}_i^*$. In all 2160 tests $\bar{w}^*$ was found with the H-GPS system that was tried first. In 33 % of the cases the target was found inside a certain region in all other cases it was found in a likely region. The binary search algorithm needed on average 18.6 steps with a standard deviation of 4.0 to find (β_1, β_2).

Algorithm 3.1 guarantees the return of a H-GPS configuration that achieves the desired performance to the specified accuracy in a finite amount of time. This is clear from the theorems and lemmas from Section 3.2. For the discrete-time version of GPS described in Section 3.1 on the contrary, no such algorithms are described in literature, to the best of our knowledge. The greatest complication is that each optimization parameter has an influence on the mean unfinished work of all three queues, which is in some cases even non-monotone. Therefore, it is necessary to resort to general techniques from gradient-free optimization, such as Nelder-Mead, simulated annealing, pattern search etc [42, 52, 107] to optimize GPS. These algorithms all produce local extrema for the defined objective function. Without extra knowledge about the behavior of GPS, it is hard to construct an algorithm that guarantees a result in finite time with the desired accuracy. For H-GPS, however, the greatest complication from GPS is absent by design, which enabled us to construct an optimization algorithm as shown in this section.

3.4 Extending to more than three classes

In this section, we briefly consider the expansion of our work to four and possibly more classes of customers. This is only a small overview of the challenges and possible solutions; more research is required. Firstly, we look at the polytope of the feasible region for the performance of work-conserving scheduling policies for four classes of customers. As a consequence of the work-conserving requirement, the sum of the mean unfinished work in all of the queues is a constant. Therefore, $\bar{w}_4 = \bar{w}_T - \bar{w}_1 - \bar{w}_2 - \bar{w}_3$ and the polytope can be drawn in 3 dimensions. An example is shown in Figure 3.14. The vertices of the polytope are the 24 possible strict priority scheduling policies, e.g. $(1 > 2 > 3 > 4), (2 > 1 > 3 > 4)$. The polytope has 8 hexagonal faces, formed by the vertices with the same class as either high or low priority class. These hexagonal faces are parallel to one of the axes or have (1,1,1) as a normal vector ($\bar{w}_4$ constant). The polytope also has 6 quadrilateral faces of which the face with the performance vector α is an example in Figure 3.14. For the face containing α, $\bar{w}_1 + \bar{w}_2$ is a constant and $\bar{w}_3 + \bar{w}_4$ is a constant. Its vertices are the strict priority policies where classes 3 and 4 have priority over 1 and 2, i.e.: $(4 > 3 > 2 > 1), (4 > 3 > 1 > 2), (3 > 4 > 1 > 2)$ and $(3 > 4 > 2 > 1)$. For every point on this face, every class-3 and class-4 customer has strict priority over customers of classes 1 and 2.

With the straightforward extension of H-GPS to four classes where on each hierarchical level exactly one queue is separated, see Figure 3.15, it is impossible to achieve all performance vectors of the polytope. For instance, the performance vector of point α cannot be achieved. There are 12 possibilities to associate the 4 classes with the 4 queues leading to different performance regions for H-GPS. With none of the 12, it is possible to achieve the performance at the point α. Indeed, with these systems, we can only reach the performance of the sides of the quadrilateral face in which α lies. This is because

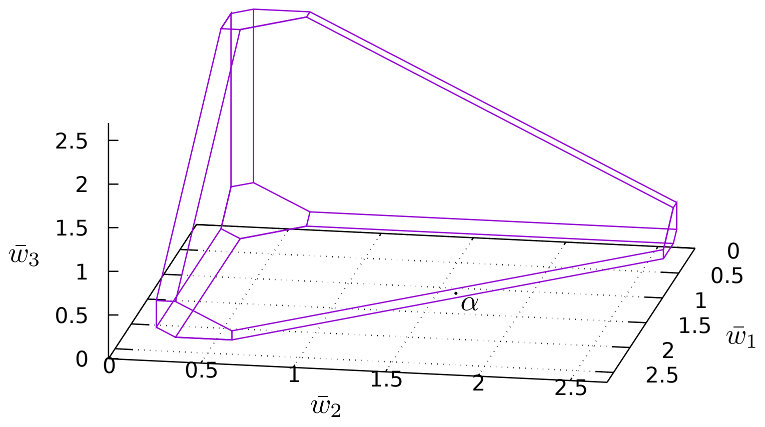


Figure 3.14: Example of the polytope Ω for a system with four classes.

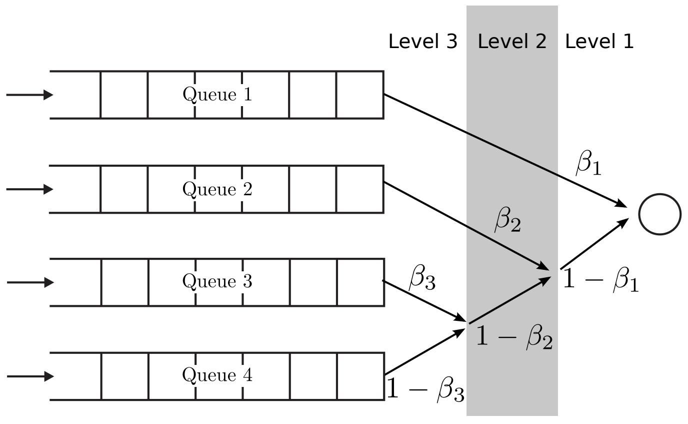


Figure 3.15: Simple extension of H-GPS for four classes.

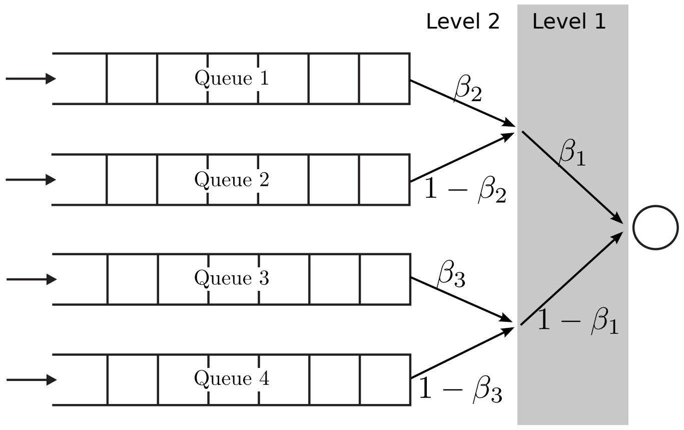


Figure 3.16: H-GPS systems built from other binary decision trees.

the parameter combinations in the H-GPS systems of the form of Figure 3.15 that give strict priority to two classes over the others are fairly limited. However, if we extend the decision tree with other hierarchical orderings of the form of Figure 3.16, we conjecture that the complete space can be achieved. In this system it is easy to give two queues priority over the other two. For instance, sending class-i customers to queue i in Figure 3.14, we give classes 1 and 2 priority over 3 and 4 by setting $\beta_1 = 0$. The proof of this conjecture is not straightforwardly drawn from the proofs presented here and possibly requires a different approach. Furthermore, we conjecture that for N classes of customers every point in the polytope Ω of possible performance vectors resulting from work-conserving policies as defined in [61] is achievable with the latter extension of H-GPS systems. Such H-GPS systems are modeled as binary decision trees to choose which queue to serve, analogously to the examples in Figures 3.15 and 3.16.

For the optimization part, the main difficulty is that intuition is lost as it is impossible to visualize the achievable regions. The number of possible binary trees quickly rises with the amount of classes. For N classes, there are $(2N-3)!! = 1{\cdot}3{\cdot}5{\cdot}\ldots{\cdot}(2N-3)$ (double factorial of odd numbers [123]) possible binary trees leading to a unique H-GPS system[2]. As such, step one of the algorithm, ordering the H-GPS

[2] The number of possible binary trees that lead to a unique H-GPS system with N classes equals the number of unordered full binary trees with N labeled leaves [123]. In a full binary tree every node has either 0 or 2 childeren. In an unordered three, the childeren have no particular order. It is clear from the definition of

systems to likeliness is hard and not scalable. We propose to use the distance measure from $\bar{\boldsymbol{w}}^*$ to the hyperplanes bounding Ω to select the first hierarchical level to optimize. For instance in the case of α in Figure 3.14, the closest hyperplane is the plane that contains α itself. As such, we optimize a two queue H-GPS system $(1+2, 3+4)$ to achieve $\bar{w}_1^* + \bar{w}_2^*$ retrieving β_1 from Figure 3.16. If this proves impossible, we should return a step and choose the hyperplane that is the second closest to $\bar{\boldsymbol{w}}^*$. In summary, the algorithm iteratively builds the binary decision tree and retraces a step when it gets blocked.

unordered full binary trees with labeled leaves that this indeed corresponds to our combinatorics problem of counting the possible number of unique H-GPS systems. The sequence looks like this starting from $N = 1$: 1, 1, 3, 15, 105, 945, 10395 …

Part II

Analysis

"Waarom gemakkelijk doen als je het ook moeilijk kunt doen."

— Simon Scharlaken

4 Two Classes

4.1 Introduction

In Part I, we studied the optimization of discrete-time GPS systems. To this end, we first determined the achievable performance vectors for the systems, so that we could use standard optimization techniques to find an achievable performance vector that minimizes the objective function. In a second step, we obtained the specific configuration of the GPS system that achieved this optimal performance vector. The algorithms we developed assumed that we could not calculate the performance measures of the system from the parameters Therefore they relied on important properties of the systems (monotonicity of the performance measures and hierarchical nature of the system) and iterative search algorithms whereby performance values were determined using simulation.

Now in Part II, we look at the analysis of the discrete-time GPS systems; in concreto, we discuss ways to calculate the performance measures from the system configuration. In this chapter, we study the two-class discrete-time GPS model from Section 1.3, i.e. with a general, but i.i.d. from slot to slot, arrival process. Specifically, we discuss the application of the power series approximation method from [145]. In the next chapter, we extend the application of this approximation method to the three-class H-GPS system we proposed in Section 3.1. Lastly, in Chapter 6, we develop a new method for the two-class model that provides a better and faster approximation; however, the method is currently limited to a bivariate Bernoulli arrival distribution.

For two customer classes, the GPS-model leads to a random walk on the two-dimensional lattice in the quarter plane. In these cases the stationary distribution can be found using the theory of boundary value problems. The formal solution, however, requires considerable numerical efforts, including the numerical determination of a conformal mapping [40, 41, 58, 145]. Other approaches for analyzing two-dimensional queueing models include the uniformization technique [88], the compensation method [4], and the power series approximation [23, 24, 76]. For a comparison of the approaches see [2].

Using power series to solve complex mathematical problems is a well known technique in queueing theory and in engineering in general [62, 128]. This power series method also goes by different names in queueing theory, such as: perturbation technique or light traffic analysis [10, 25, 57, 91, 92, 126]. Most power series approximations are based on an expansion of the steady state probabilities as a power series in the parameter denoting the total load offered to the system. Using the balance equations of the system, subsequently, the coefficients of the power series can be calculated iteratively. The result is that the accuracy of the approximation deteriorates as the load increases. In [145], Walraevens et al. study the two-class discrete-time GPS system we also study in this dissertation[1] using a power series approximation technique. The great novelty from that paper is that the power series is constructed in the parameter β rather than in the load. In the next section, we summarize the main elements from [145] that will be used in the remainder of this dissertation. In the rest of this chapter, we will discuss the difficulties and some new adjustments in applying the power series approximation from [145].

4.2 Power series approximation of the joint probability generating function

In this section, we summarize the power series approximation from [145] adapted to the setting of this dissertation, i.e. by only considering single slot service times for the customers. The reader should note that the author of this dissertation was not involved in the writing nor the research that lead to [145], and thus the results from this Section 4.2. We merely summarize the content here, to make the text

[1] albeit for customers requiring geometric service times

self-contained as we will heavily build upon this paper in the coming sections and the next chapter.

For clarity, we repeat the most important definitions. In the two-class system, when both classes are backlogged, class 1 (class 2) is served with probability β $(1-\beta)$. In all other cases the backlogged class (if any), is served. We recall that $w_{j,k}$ denotes the unfinished work of class j in slot k, and $\boldsymbol{w}_k = (w_{1,k}, w_{2,k}, w_{3,k})$. Analogously $a_{j,k}$ and $\boldsymbol{a}_k = (a_{1,k}, a_{2,k}, a_{3,k})$ are used to denote the arrivals in slot k. Lastly, we repeat the system equations of (1.1).

- if $\boldsymbol{w}_k = \mathbf{0}$
$$\boldsymbol{w}_{k+1} = \boldsymbol{a}_k$$
- if $w_{i,k} > 0$; $w_{3-i,k} = 0$ with $i = \{1,2\}$
$$\begin{aligned} w_{i,k+1} &= w_{i,k} - 1 + a_{i,k} \\ w_{3-i,k+1} &= a_{3-i,k} \end{aligned}$$
- if $w_{j,k} > 0$ with $j = \{1,2\}$
$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (1,0) + \boldsymbol{a}_k \quad \text{w.p.} \quad \beta \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,1) + \boldsymbol{a}_k \quad \text{w.p.} \quad 1-\beta \end{aligned}$$

We define the joint probability generating function of the unfinished work in slot k as $W_k(z_1, z_2) = \mathrm{E}[z_1^{w_{1,k}} z_2^{w_{2,k}}]$. Using this definition, we transform the system equations to the following functional equation of pgf's:

$$\begin{aligned} &W_{k+1}(z_1, z_2) \qquad (4.1) \\ &= A(z_1, z_2) \begin{pmatrix} W_k(0,0) + \frac{1}{z_2}\Big(W_k(0,z_2) - W_k(0,0)\Big) \\ +\frac{1}{z_1}\Big(W_k(z_1,0) - W_k(0,0)\Big) \\ +\left(\frac{\beta}{z_1} + \frac{1-\beta}{z_2}\right) \begin{pmatrix} W_k(z_1,z_2) - W_k(0,z_2) \\ -W_k(z_1,0) + W_k(0,0) \end{pmatrix} \end{pmatrix}. \end{aligned}$$

Next, with the stability condition ($\lambda_T < 1$) fulfilled, we let $k \to \infty$, with $W_k(z_1,z_2) = W_{k+1}(z_1,z_2) = W(z_1,z_2)$ and $W(z_1,z_2)$ the steady state pgf. The previous expression thus translates to the functional equation for the steady state pgf of the unfinished work.

$$\begin{aligned} &W(z_1,z_2)\Big(z_1(z_2 - A(z_1,z_2)) - A(z_1,z_2)\beta(z_2 - z_1)\Big) \qquad (4.2) \\ &= A(z_1,z_2) \begin{pmatrix} W(z_1,0)(1-\beta)(z_2 - z_1) \\ -W(0,z_2)\beta(z_2 - z_1) \\ +W(0,0)\left(z_2(z_1 - 1) + \beta(z_2 - z_1)\right) \end{pmatrix} \end{aligned}$$

Now, we need to eliminate the boundary functions $W(z_1, 0)$, $W(0, z_2)$ and the constant $W(0, 0)$. The great difficulty is that the functional equation includes both $W(z_1, 0)$ and $W(0, z_2)$, whereas for a strict priority scheme the functional equation for the steady state pgf only comprises one of them [143][2]. We, however, observe that $W(0, z_2)$ only appears with a factor β. By expanding the pgf's as power series in β and equating corresponding coefficients to iteratively calculate these coefficients of the power series, one unknown is eliminated.

Assuming $W(z_1, z_2)$ is an analytic function of β in a neighborhood of 0, we write $W(z_1, z_2) = \sum_{m=0}^{\infty} V_m(z_1, z_2)\beta^m$ as its power series expansion in β for all z_1 and z_2 in the unit disk. Substituting the power series in (4.2) and equating coefficients of corresponding powers of β, results in a functional equation for V_m, for $m \geq 0$:

$$V_m(z_1, z_2)\Big(z_1(z_2 - A(z_1, z_2))\Big) = A(z_1, z_2)\begin{pmatrix} (z_2 - z_1)V_m(z_1, 0) + z_2(z_1 - 1)V_m(0, 0) \\ +(z_2 - z_1)P_{m-1}(z_1, z_2) \end{pmatrix}, \tag{4.3}$$

with $P_m \triangleq V_m(z_1, z_2) - V_m(z_1, 0) - V_m(0, z_2) + V_m(0, 0)$, $m \geq 0$ and $P_{-1} \triangleq 0$. For a certain fixed m, we assume that $P_{m-1}(z_1, z_2)$ is known and we want to express $V_m(z_1, z_2)$ in terms of $P_{m-1}(z_1, z_2)$. We then, indeed, see that two unknowns are left $V_m(z_1, 0)$ and $V_m(0, 0)$. By using Rouché's theorem [3], we prove that $z_2 - A(z_1, z_2)$ has one zero in the unit disk of z_2 for an arbitrary z_1 in the unit disk. Denote this zero by $Y(z_1)$; it is defined in the unit disk as $Y(z_1) - A(z_1, Y(z_1)) = 0$ and $|Y(z_1)| < 1$. The implicit function theorem then says that $Y(z_1)$ is an analytic function in the unit disk. Since $W(z_1, z_2)$ is analytic for all z_1 and z_2 in the unit disk, the $V_m(z_1, z_2)$ are as well. Therefore, the right-hand side of (4.3) should equal zero for $z_2 = Y(z_1)$. This gives

$$V_m(z_1, 0) = -\frac{Y(z_1)(z_1 - 1)}{Y(z_1) - z_1} V_m(0, 0) - P_{m-1}(z_1, Y(z_1)). \tag{4.4}$$

By substituting (4.4) in (4.3), we obtain

$$V_m(z_1, z_2)\Big(z_1(z_2 - A(z_1, z_2))\Big) = A(z_1, z_2)\begin{pmatrix} (z_2 - z_1)Q_{m-1}(z_1, z_2) + V_m(0, 0)\cdot \\ \left(z_2(z_1 - 1) - \frac{Y(z_1)(z_1-1)(z_2-z_1)}{Y(z_1)-z_1}\right) \end{pmatrix}, \tag{4.5}$$

[2] This can also easily be seen by setting $\beta = 0$ or 1 in (4.2).

with

$$\begin{aligned}Q_m(z_1,z_2) &\triangleq P_m(z_1,z_2) - P_m(z_1,Y(z_1))\\ &= V_m(z_1,z_2) - V_m(z_1,Y(z_1)) - V_m(0,z_2) + V_m(0,Y(z_1)).\end{aligned} \tag{4.6}$$

The last step is the calculation of $V_m(0,0)$, this constant is found from the normalization condition. Since $W(1,1) = 1$ for all β, it follows that $V_0(1,1) = 1$ and $V_m(1,1) = 0$ for all $m > 0$. Setting $z_1 = z_2 = 1$ in (4.5) and using that $Q_m(1,1) = 0$ for all $m \geq 0$, we find that $V_0(0,0) = 1 - \lambda_T$ and $V_m(0,0) = 0$ for $m > 0$. We finally arrive at the following relations for $m > 0$:

$$V_m(z_1,z_2) = \frac{A(z_1,z_2)(z_2 - z_1)Q_{m-1}(z_1,z_2)}{z_1(z_2 - A(z_1,z_2))} \tag{4.7}$$

and

$$V_0(z_1,z_2) = \frac{(1-\lambda_T)A(z_1,z_2)(z_1-1)(z_2-Y(z_1))}{(z_2 - A(z_1,z_2))(z_1 - Y(z_1))}. \tag{4.8}$$

Hence starting from V_0 in (4.8), every V_m can, in theory, be determined iteratively via (4.7) and (4.6).

The implicit function $Y(z_1)$

We defined the function $Y(z_1)$ implicitly as $Y(z_1) = A(z_1, Y(z_1))$, i.e. the zeros for arbitrary z_1 in the unit disk of $z_2 - A(z_1,z_2)$. We show that $Y(z_1)$ is actually the pgf of the stochastic variable y, which is defined as the number of class-1 arrivals during a sub-busy period initiated by a random customer in the system with $\beta = 0$ (i.e. in a strict priority system with priority for class-2). This was shown in [141] and we merely summarize here. A sub-busy period is initiated by the arrival of a random customer, this customer is subsequently *tagged*. At the beginning of the arrival slot of the tagged customer, we count x customers that need to be served before the tagged customer. The sub-busy period ends when the amount of customers that need to be served before the tagged customer is for the first time less than x, i.e. $x - 1$.

If the tagged customer is of class-2, the sub-busy period is only 1 slot long. If the tagged customer is of class-1 and in the first slot of this sub-busy period no class-2 customers (having priority with $\beta = 0$) arrive, then none of the arrivals in that slot will cut in line before the tagged customer. Subsequently, the amount of customers needing service before the tagged customer decreases to $x - 1$ and the sub-busy period ends. In that case $y = a_1^{(1)}$, whereby we denote the number of class-j arrivals in the k-th slot of the sub-busy period by $a_j^{(k)}$. If in the first slot class-2 customers arrive, they need to be served before the tagged customer. The amount of customers needing service before the tagged customer thus does not decrease, it is $x - 1 + a_2^{(1)}$ and $a_2^{(1)} > 0$. In fact every class-2 customer initiates a new sub-busy period, each having the same distribution as the original sub-busy period. For y we thus need to count the class-1 arrivals in the first slot of the sub-busy period and add the class-1 arrivals in the newly begun sub-busy periods. We thus write:

$$y = a_1^{(1)} + \sum_{l=1}^{a_2^{(1)}} y_l^{(1)}, \tag{4.9}$$

whereby we denoted the amount of class-1 arrivals in the sub-busy period initiated by the l-th class-2 arrival in the first slot of the sub-busy period of the tagged customer by $y_l^{(1)}$. Taking the z-transform of (4.9) and using the fact that (i) each $y_l^{(1)}$ is independent and identically distributed as y itself and (ii) $y_l^{(1)}$ is independent from $a_j^{(1)}$ $(j = 1, 2)$, we find

$$Y(z_1) = A(z_1, Y(z_1)).$$

For $\beta = 0$, the second queue has strict priority over the first. The pgf of the unfinished work in this case equals $W(z_1, z_2) = V_0(z_1, z_2)$ as given in (4.8). This last expression is indeed the pgf in a discrete-time priority queueing system with the first queue having low priority (see [143]).

Subsequently, we calculate the performance measures of the system:

$$\bar{w}_j(\beta) = \left.\frac{\partial W(z_1, z_2)}{\partial z_j}\right|_{z_1=z_2=1} = \sum_{m=0}^{\infty} \beta^m \left.\frac{\partial V_m(z_1, z_2)}{\partial z_j}\right|_{z_1=z_2=1}.$$

As $\bar{w}_1(\beta) + \bar{w}_2(\beta)$ equals a constant $\bar{w}_T$, we find that for $m > 0$

$$\left.\frac{\partial V_m(z_1, z_2)}{\partial z_1}\right|_{z_1=z_2=1} = - \left.\frac{\partial V_m(z_1, z_2)}{\partial z_2}\right|_{z_1=z_2=1}.$$

Consequently, we focus on just one of both, for instance, $\bar{w}_2$, as $\bar{w}_1$ can be calculated as $\bar{w}_T - \bar{w}_2$. Furthermore, if we note $\bar{w}_2(\beta) = \mathrm{E}[w_2] = \sum_{m=0}^{\infty} \mathrm{E}[w_2]_m \beta^m$, we write

$$\mathrm{E}[w_2]_m = \left.\frac{\partial V_m(z_1, z_2)}{\partial z_2}\right|_{z_1=z_2=1}.$$

A first approximation constitutes the truncation of this power series

$$\bar{w}_2(\beta) \approx \sum_{m=0}^{M} \mathrm{E}[w_2]_m \beta^m.$$

Clearly, the larger M the better the approximation gets[3]. The approximation is good for β in the neighborhood of 0, the expansion

[3]The bigger M already is, the smaller the gain of adding an extra coefficient is though.

point of the power series. Evidently, this is because the approximation matches the first M derivatives in $\beta = 0$.

Additionally, we recall that the problem is symmetric in β, as noted in Section 1.3. We can thus reuse the expressions to calculate the coefficients of a power series in $\beta = 1$. This is done by replacing $A(z_1, z_2)$ by $A(z_2, z_1)$, i.e., sending class-1 customers to queue 2 and vice versa. To subsequently calculate the mean class-2 unfinished work, we need to use the formulas for queue 1. We note V_m with $A(z_1, z_2)$ replaced by $A(z_2, z_1)$ as $\tilde{V}_m$ and

$$\mathrm{E}[\tilde{w}_2]_m = \left.\frac{\partial \tilde{V}_m(z_1, z_2)}{\partial z_1}\right|_{z_1=z_2=1}.$$

Then, we obtain

$$\bar{w}_2(\beta) = \sum_{m=0}^{\infty} \mathrm{E}[\tilde{w}_2]_m (1-\beta)^m.$$

A truncation of this power series provides a good approximation in the neighborhood of $\beta = 1$.

Obviously, our goal is to find a good approximation for the whole domain $\beta \in [0, 1]$. An alternative for the truncated power series (polynomials) are Padé approximants, which are rational functions. We explain their definition and use in the next section.

4.3 Padé approximants

Padé approximants are rational functions of the form

$$[L/K](\beta) = \frac{\sum_{l=0}^{L} c_{1,l}\beta^l}{\sum_{k=0}^{K} c_{2,k}\beta^k}. \tag{4.10}$$

These have $L + K + 2$ parameters of which we fix one, $c_{1,0} = 1$, as a normalization[4]. The remaining $L+K+1$ parameters need to be set to make for a good approximation. In our case, we calculate them from the coefficients of the power series, by requiring that the derivatives of the Padé approximant in $\beta = 0$ and 1 equal the derivatives of the power series of the performance measures in $\beta = 0$ and 1. Say, we calculated both power series truncated at the M-th order coefficient, then this results in $2(M+1)$ different functions ($L = 0, \ldots, 2M+1$) of which the $[L/0]$ approximant represents a polynomial approximation.

[4]Otherwise you get an identical function by dividing every parameter by $c_{1,0}$.

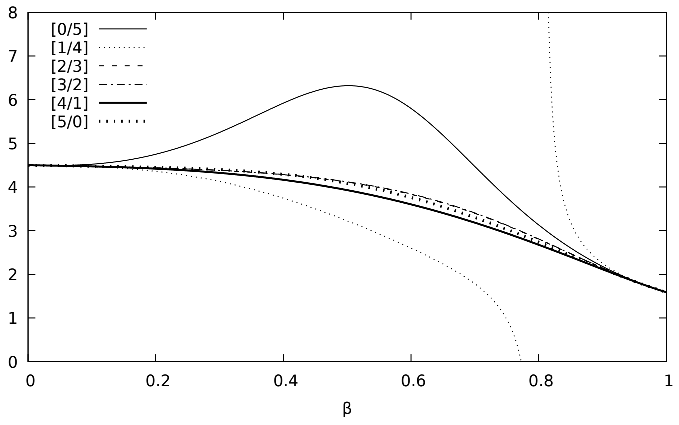


Figure 4.1: Padé approximants for $\bar{w}_1(\beta)$ for a binomial arrival process with $N = 16$, $\alpha = 0.8$ and $\lambda_T = 0.9$

The Padé approximants are exact in $\beta = 0$ and $\beta = 1$ and accurate in their neighborhoods by design (matching the first M derivatives in both points). We expect the Padé approximants to be less accurate moving further away from the endpoints 0 and 1.

Some observations of the Padé approximants even indicate greater challenges in using it reliably as an approximation. In Figure 4.1, we show an example of the Padé approximants to illustrate these observations. For the example, a two-dimensional binomial arrival process is used with $N = 16$, $\alpha = 0.8$ and $\lambda_T = 0.9$. The figure shows the Padé approximants for $\bar{w}_1$ with the power series truncated at $M = 2$.

The first observation is that, since Padé approximants are rational functions, they can have singularities in the interval $[0, 1]$. This could be a reason to stick with polynomials as these cannot have any singularities, but our results show that at least one of the Padé approximants always leads to a better approximation. The difficulty is to filter out the *good* approximations from the *bad*. For instance, the [1/4] Padé approximant of $\bar{w}_1(\beta)$ in Figure 4.1 has a pole for $\beta = 0.8$. Obviously, the actual function $\bar{w}_j(\beta)$ cannot have such poles as the system is assumed to be stable for all β and is shown to be continuous in Theorem 2.1. This means that we can, a priori, discard the approximants which have poles in $[0, 1]$.

A second observation is that some of the approximants are non-monotone. An example can also be seen in Figure 4.1; the [0/5] Padé approximant has a maximum. We however proved in Theorem 2.1

that $\bar{w}_1(\beta)$ is monotonically decreasing and $\bar{w}_2(\beta)$ is monotonically increasing. We can thus also filter out these approximants by calculating the zeros of the first derivative: zeros between 0 and 1 indicate a local extremum and therefore non-monotonicity.

Last but not least, we know from Corollary 2.1 that

$$\bar{w}_1(\beta) + \bar{w}_2(\beta) = \bar{w}_T. \tag{4.11}$$

As mentioned before $\bar{w}_T$ can be calculated as the mean unfinished work of a single-class system, aggregating the two classes of customers in one queue [32]. We, however, observe that calculating approximations for $\bar{w}_1(\beta)$ and $\bar{w}_2(\beta)$ independently does not sum to $\bar{w}_T$, and in many cases is far off. This means we can double the amount of available approximants by calculating $\bar{w}_j(\beta)$ from $\bar{w}_{3-j}(\beta)$ and $\bar{w}_T$, by using (4.11). For instance, if we calculate the M-th order approximation for $W(z_1, z_2)$, we can calculate $2(M+1)$ parameters for $\bar{w}_j(\beta)$, i.e. $2(M+1)$ Padé approximants for $\bar{w}_1(\beta)$ and $2(M+1)$ approximants for $\bar{w}_2(\beta)$. This leads to $4(M+1)$ approximations for $\bar{w}_1(\beta)$, $2(M+1)$ Padé approximants and $2(M+1)$ calculated using the approximants of $\bar{w}_2(\beta)$, i.e. $\bar{w}_T - \bar{w}_2(\beta)$. Most of these approximations are different except for both polynomial approximants, which already satisfy (4.11) (this can easily be seen from the way they are constructed). As an example, we have the 6 approximants for $\bar{w}_1(\beta)$ from Figure 4.1. By calculating the 6 Padé approximants for $\bar{w}_2(\beta)$ (from the same truncated power series for the joint pgf) and using (4.11), we find another 6 approximants for $\bar{w}_1(\beta) = \bar{w}_T - \bar{w}_2(\beta)$. These are depicted in Figure 4.2.

After filtering the invalid solutions (non-monotonic functions, functions with singularities both in $[0, 1]$), we end up with a set of approximations for $\bar{w}_j(\beta)$. The best approximation from this set depends on the parameters of the studied system. When no simulation is available (which is obviously the case when one wants to use the approximations in practice) it is consequently, not a priori, clear which approximation performs best. Therefore, we propose as a heuristic to aggregate the set of approximations by taking the average over the set. By taking the average over this set of approximations, that all have the same M derivatives in the endpoints, these derivatives in the endpoints are retained in the resulting average. Taking all this together, we get Algorithm 4.1.

It should be noted that the expressions obtained for the Padé approximants are symbolic and that they do not have to be recalculated for each set of parameters. Filtering the approximations for singularities and non-monotonicity should be done for each parameter set as the appearance of these phenomena depends on the values for the

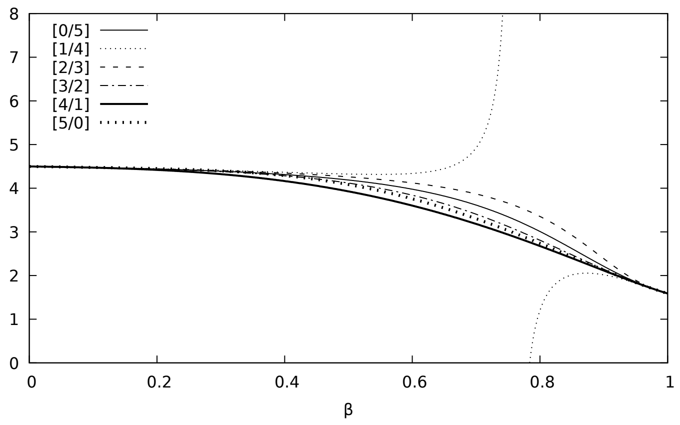


Figure 4.2: $\bar{w}_1(\beta)$ calculated as $\bar{w}_T - \bar{w}_2(\beta)$ for a binomial arrival process with $N = 16$, $\alpha = 0.8$ and $\lambda_T = 0.9$, with $\bar{w}_2(\beta)$ a Padé approximant.

```
   /* Aggregate queues and use single-class results */
1  $\bar{w}_T$ = calc_total_unf_work(arrival_process);
   /* Calculate the truncated power series for $\bar{w}_j(\beta)$ to $M$-th
      order */
2  $\bar{w}_1$_left = calc_unf_work(1, arrival_process, M, $\beta = 0$);
3  $\bar{w}_1$_right = calc_unf_work(1, arrival_process, M, $\beta = 1$);
4  $\bar{w}_2$_left = calc_unf_work(2, arrival_process, M, $\beta = 0$);
5  $\bar{w}_2$_right = calc_unf_work(2, arrival_process, M, $\beta = 1$);
   /* Construct Padé approximants using the coefficients of the
      power series */
6  list_pade_$\bar{w}_1$ = padé($\bar{w}_1$_left,$\bar{w}_1$_right);
7  list_pade_$\bar{w}_2$ = padé($\bar{w}_2$_left,$\bar{w}_2$_right);
   /* Use (4.11) to construct extra approximations */
8  concatenate(list_pade_$\bar{w}_1$, $\bar{w}_T$-list_pade_$\bar{w}_2$);
   /* Remove approximants with poles and non-monotonicities */
9  filter_poles(list_pade_$\bar{w}_1$);
10 filter_non_monotone(list_pade_$\bar{w}_1$);
   /* Average remaining approximants */
11 return average(list_pade_$\bar{w}_1$);
```

Algorithm 4.1: Filter and aggregate the Padé approximants to get one master approximation.

parameters. We can thus use the approximation to do sensitivity analysis on the parameters of the system without the need for large amounts of computations (e.g. simulations).

4.4 Using the approximants for optimization

In this section, we use the approximation from Algorithm 4.1 of the previous section and apply it in an optimization setting. Specifically, we study the minimization of the cost function

$$F(\beta,\gamma) = \gamma\Big(\bar{d}_1(\beta)\Big)^n + (1-\gamma)\Big(\bar{d}_2(\beta)\Big)^n.$$

As mentioned in Chapter 2, for single slot service times we can write the mean delay as a function of the mean unfinished work (which equals the mean queue content for single slot service times) using Little's law, i.e., $\bar{d}_j(\beta) = \frac{\bar{w}_j(\beta)}{\lambda_j}$. Subsequently, we observe that $F(\beta,\gamma)$ is of the form of the objective function which was discussed in Theorem 2.3, with $g_j(x) = \left(\frac{x}{\lambda_j}\right)^n$ increasing and convex functions. The theorem then stated that the minimum of F is reached at a β value different from 0 or 1 when γ is in the interval $]\phi(0),\phi(1)[$. Furthermore, $\beta_{\text{opt}}(\gamma)$ is the inverse of $\phi(\beta)$, that equals for this particular choice of F:

$$\phi(\beta) = \frac{\lambda_1^n\Big(\bar{w}_2(\beta)\Big)^{n-1}}{\lambda_2^n\Big(\bar{w}_1(\beta)\Big)^{n-1} + \lambda_1^n\Big(\bar{w}_2(\beta)\Big)^{n-1}}.$$

Taking the n-th derivative of $\phi(\beta)$, we see that only derivatives up to the M-th of $\bar{w}_j(\beta)$ appear in the resulting expression. As a consequence, if we know the exact values of up to the M-th derivative of $\bar{w}_j(\beta)$, we can also calculate the exact values of up to the M-th derivative of $\phi(\beta)$. By using the chain rule and taking the derivative of both sides of the equation $\phi^{-1}(\phi(\beta)) = \beta$, we find $\mathcal{D}(\phi^{-1})(\phi(\beta))\,\mathcal{D}\,\phi(\beta) = 1$ and thus $\mathcal{D}(\phi^{-1})(\phi(\beta)) = (\mathcal{D}\,\phi(\beta))^{-1}$. We can thus easily see that having up to the M-th derivative of ϕ in $\beta = 0$ and 1 also gives us up to the M-th derivative of $\beta_{\text{opt}} = \phi^{-1}$ in $\phi(0)$ and $\phi(1)$.

For the cost function, we need approximations for both $\bar{w}_1(\beta)$ and $\bar{w}_2(\beta)$. Our tests have shown that a better approximation for $\phi(\beta)$ (and thus $\beta_{\text{opt}}(\gamma)$) is found when calculating the approximant for one of the queues and using (4.11) to calculate the approximant for the complementing queue, as opposed to calculating the approximants for both queues independently. Still there are two options to construct an approximation of $\beta_{\text{opt}}(\gamma)$. The first one is to use $\bar{w}_2(\beta)$ constructed with Algorithm 4.1 and use that in $\phi(\beta)$; after inversion we then obtain an approximation for $\beta_{\text{opt}}(\gamma)$. A second option is to use each $\bar{w}_2(\beta)$ approximant in the filtered list constructed in line 10 of Algorithm 4.1 and construct an approximation for $\phi(\beta)$. Subsequently, the list of $\phi(\beta)$ is averaged and afterwards inverted to obtain an approximation for $\beta_{\text{opt}}(\gamma)$. We summarized this in Algorithm 4.2. Only in very specific cases both construction methods lead to the same approximation. We have found that in general the second method provides better results as we will demonstrate at the end of this section.

```
  /* Construct a list of filtered Padé approximants cf. lines
     1-10 from Algorithm 4.1                                   */
1 list_pade_$\bar{w}_1$ = filtered_pade_list(arrival_process,M);
  /* Construct a list of $\phi(\beta)$ approximants based on the Padé
     approximants                                              */
2 list_$\phi$ = phi(list_pade_$\bar{w}_1$,$\bar{w}_T$);
  /* Average the $\phi(\beta)$ approximants                    */
3 avg_$\phi$ = average(list_$\phi$);
  /* Return the inverse of the averaged approximant for $\phi(\beta)$   */
4 return invert(avg_$\phi$);
```

Algorithm 4.2: Approximation for $\beta_{\text{opt}}(\gamma)$ constructed using the Padé approximants from Algorithm 4.1.

4.4.1 Influence of the arrival process

First, we study the influence of the arrival process on the approximation of $\beta_{\text{opt}}(\gamma)$. To this end, we fix $n = 2$ in the cost function and study two arrival processes for several configurations. The first arrival process is a two-dimensional binomial arrival process as specified in Section 1.6. We plotted some results for various parameters of the arrival process in Figure 4.3. The leftmost figures display the performance of increasing orders (M) of the approximation for $\beta_{\text{opt}}(\gamma)$ by plotting them together with results from simulation (obtained using the specifics detailed in Section 1.5). In the remainder of this section, we will denote the real optimal β as β_{opt}, this value is obtained by simulation. Approximations for the optimal β are denoted by $\tilde{\beta}_{\text{opt}}$, these values are obtained by using Algorithm 4.2. In the center figures, we have plotted the value of the cost function for β_{opt}

as a function of γ with a solid line. The other curves are the real costs that would be obtained when using the suboptimal $\tilde{\beta}_{\text{opt}}$ value found from the respective approximations. The graphs on the right show the fractional difference between the approximation curves and the simulated curve in the center figures. This fractional difference is calculated as

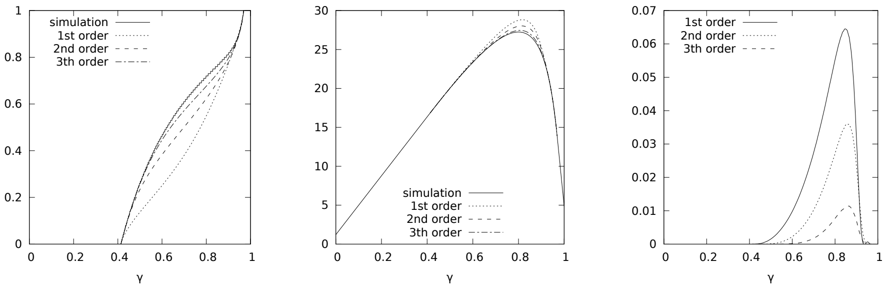
$$\frac{\left|F(\tilde{\beta}_{\text{opt}}) - F(\beta_{\text{opt}})\right|}{F(\beta_{\text{opt}})}.$$

As can be seen from the figures, increasing the order of the approximation M always leads to a better approximation of the real results. Specifically, we see that the approximation at the endpoints (β_{opt} equals 0 or 1) is progressively better. This could be expected by the way we constructed the approximation and the properties of the Power Series Approximation. It was already shown analytically and we now visually confirm that increasing the order matches more derivatives of $\beta_{\text{opt}}(\gamma)$ at the endpoints. The required extra computational effort thus pays off.

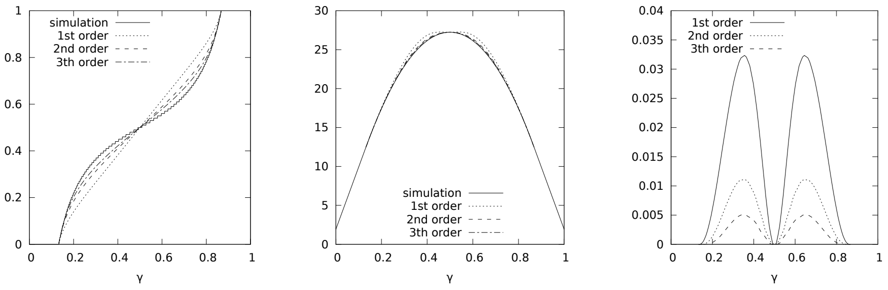
We also see that misprediction of β_{opt} does not lead to the same level of misprediction of the cost. For instance, we see from the left graph of Figure 4.3a that the maximum misprediction of the optimal weight by the first order approximation is 33% at $\gamma = 0.7$. Looking at the right graph, we however see that the misprediction of the optimal cost is about 3%. We see a maximum misprediction of the optimal cost of 6.5% for $\gamma = 0.85$ where the error on β_{opt} is 18%. We conclude that even with a far from optimal weight value we can get close to the optimal cost.

One other conclusion is that a higher load on the system reduces the accuracy of the approximation. The error (with respect to the real cost) for a load of 90% is at its worst about 1% using the third order approximation, but for a load of 70% the approximation is nearly perfect. Even when only the first order approximation is used the fractional difference is less than 0.2%. Using the third order approximation, we got an error of 30% for a load of 99% and for a load of 50% we had a maximum error of 0.05%. This follows from the fact that with a lower load there is less queueing, this results in lower values for $\bar{w}_j(\beta)$. As a consequence the difference between $\phi(0)$ and $\phi(1)$ is smaller, leaving less room (and thus error) for possible functions in between.

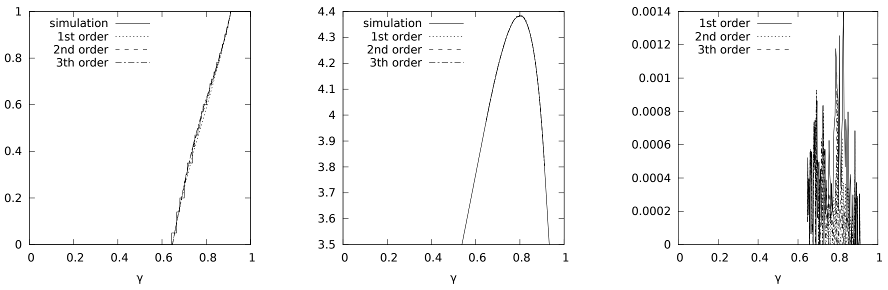
The second arrival process we used to analyze the performance of the approximation applied to optimization (still for $n = 2$), is a



(a) $n = 2$, $N = 16$, $\alpha = 0.8$, $\lambda_T = 0.9$



(b) $n = 2$, $N = 16$, $\alpha = 0.5$, $\lambda_T = 0.9$



(c) $n = 2$, $N = 16$, $\alpha = 0.8$, $\lambda_T = 0.7$

Figure 4.3: $\beta_{\text{opt}}(\gamma)$ (left), real F when using approximation (center), fractional difference (right) for several parameter combinations.

bivariate Bernoulli process with pgf

$$A(z_1, z_2) = 1 - \lambda_1 - \lambda_2 + \rho + (\lambda_1 - \rho)z_1 + (\lambda_2 - \rho)z_2 + \rho z_1 z_2.$$

From this pgf, we see that λ_j is again the arrival rate of class j in accordance with previous definitions. ρ is the probability that packets from both classes arrive in the same slot. No more than two packets can arrive in the same slot using this arrival process. If $\rho = 0$ then there is at most one arrival per slot; as a consequence this packet can be served in the slot thereafter and no backlog occurs.

We always used the third order approximation for tests with the bivariate Bernoulli arrival process. Figure 4.4 shows the mean absolute error on β_{opt} of the approximation. This mean is calculated by sampling over γ. For each γ, we calculate the absolute difference between β_{opt} from simulation and $\tilde{\beta}_{\text{opt}}$. Subsequently, the mean is calculated over all samples. Visually it is the mean vertical distance between the simulated curve and an approximated curve for β_{opt} in the left graph of Figure 4.3. The mean fractional error on F when $\tilde{\beta}_{\text{opt}}$ is used, is shown in Figure 4.5. Visually this is the mean value for one of the curves in the rightmost graph in Figure 4.3, obtained by sampling. We did this experiment for $\alpha \triangleq \frac{\lambda_1}{\lambda_T} = 0.2$ and $\alpha = 0.5$, whereby ρ was kept equal to $\frac{\lambda_1}{2}$. The same conclusions as in the previous paragraph can be drawn. A higher load does not increase the misprediction on β_{opt} whereas it does increase the relative induced error on F. For one dataseries, we also included the standard deviation over the error samples as error bars in the graph. As the load increases so does the standard deviation. We only show error bars for one series, as to not overload the figure, but the same trend is seen for the other series. This all proves that the approximation is less accurate for higher loads, not only on average, there is also a higher variability on the error.

To test the robustness of the approximation, we ran a more involved simulation. The bivariate Bernoulli arrival process has three parameters $(\lambda_1, \lambda_2, \rho)$. We ran 108 separate simulations taking samples out of the possible parameter space for the arrival process. Our program chose λ_T uniformly in $[0.3, 0.99]$ (excluding very low loads), subsequently α in $[0.01, 0.99]$ and lastly ρ in $]0, \min(\lambda_1, \lambda_2)]$ (excluding 0 as there is never any backlog in that situation). From these parameters the conversion to the parameters $\lambda_1, \lambda_2, \rho$ is evident.

For each sample point, we also calculated the minimum of F using the third order approximation constructed with Algorithm 4.2 (i.e. with $M = 3$). Subsequently, the mean absolute error and standard deviation on the difference between β_{opt} and $\tilde{\beta}_{\text{opt}}$ were calculated.

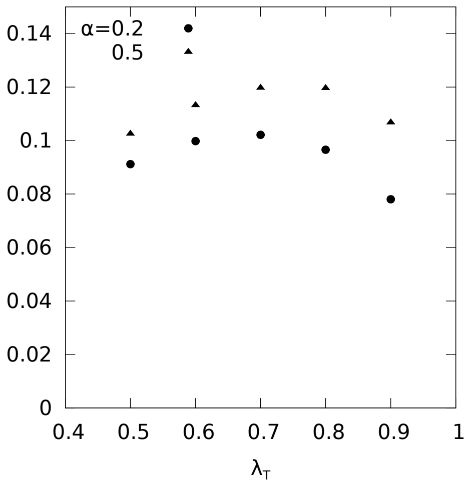


Figure 4.4: Mean absolute error on β_{opt} for bivariate Bernoulli arrival process with increasing load.

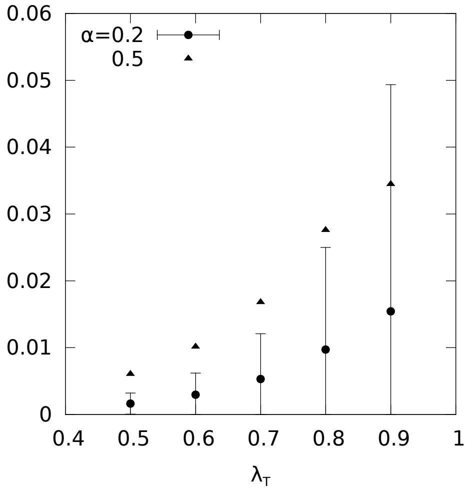


Figure 4.5: Mean fractional error on F by using approximated β_{opt} for bivariate Bernoulli arrival process with increasing load.

Table 4.1: Performance of the approximation over 108 random samples for the parameters of the second arrival process.

			Over all samples	
			Mean	STDEV
Over all γ	$\left\|\tilde{\beta}_{\text{opt}} - \beta_{\text{opt}}\right\|$	Mean	0.085	0.025
		STDEV	0.053	0.019
	$\frac{\left\|F(\tilde{\beta}_{\text{opt}}) - F(\beta_{\text{opt}})\right\|}{F(\beta_{\text{opt}})}$	Mean	0.89%	1.12%
		STDEV	1.77%	2.67%

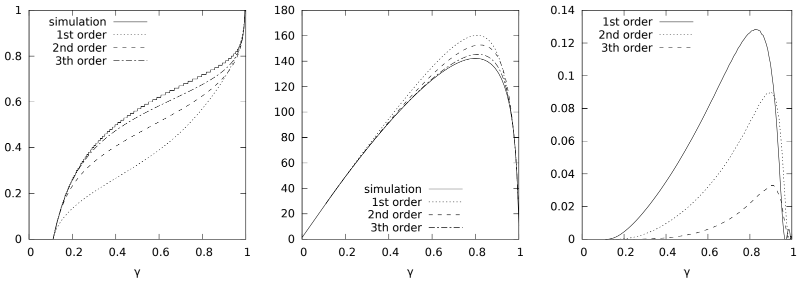
Secondly, we calculated the fractional difference between the value of the cost function at β_{opt} and the value of the cost function at $\tilde{\beta}_{\text{opt}}$ from the approximation. Lastly, we calculated the mean and standard deviation over all samples for these values, they are displayed in Table 4.1. For instance, the top left cell 0.085 is the mean (column) over all 108 samples of the means (row) per sample over γ; the top right cell 0.025 is the standard deviation (column) over the 108 samples of the means (row) per sample over γ. We can see that the approximation performs well with a mean fractional error on F of less than 1% and a mean standard deviation of less than 2%.

Each of the 108 simulations took about 3 hours on a modern processor (using a single core). The calculations for the Padé approximants are done in less than 5 minutes and can be reused for each sample (lines 1-8 of Algorithm 4.1). In fact, it can be reused by everyone for all possible parameters and arrival processes, so it really only needs to be calculated *once*. Filtering the approximations (lines 9-10 of Algorithm 4.1) and subsequently averaging and inverting (Algorithm 4.2) needs to be done for each sample, but this is done instantly. The time gains from using the approximation are self-evident.

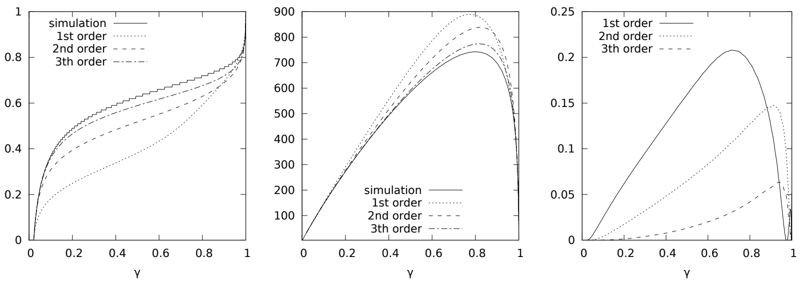
4.4.2 Influence of the cost function

Now we look at the influence of the cost function for a given arrival process; specifically we vary the exponent n in the cost function $F(\beta, \gamma)$ specified earlier. With $n > 1$, we keep the g functions (in the sense of Theorem 2.3) convex as to yield optimal weights differing from 0 or 1. These results are given in Figure 4.6 in the same format as in Figure 4.3.

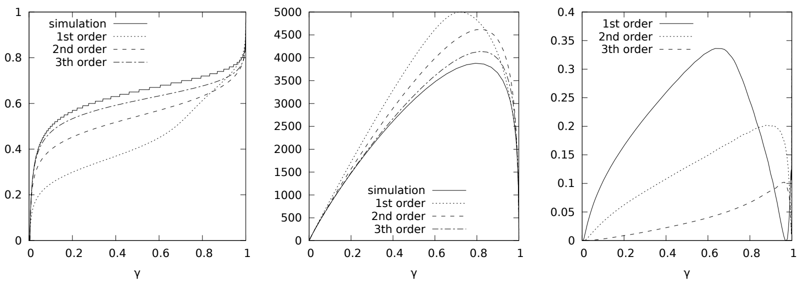
We see that with higher powers of the average delay (i.e.: higher n) the amount of misprediction of β_{opt} stays roughly the same, but the fractional cost difference increases. This can be explained by



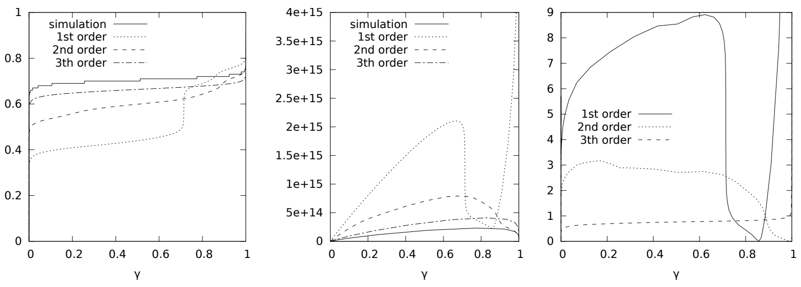
(a) $n = 3$



(b) $n = 4$



(c) $n = 5$



(d) $n = 20$

Figure 4.6: $\beta_{\text{opt}}(\gamma)$ (left), real F when using approximation (center), fractional difference (right) for several cost functions. The arrival process uses the following parameters: $N = 16$, $\alpha = 0.8$, $\lambda_T = 0.9$.

making the following reasoning. Take two different cost functions one (F_1) with a low power, say $n = 2$, and one (F_2) using a high power, for instance $n = 20$. If the approximations for both cost functions indicate the same β_{opt} for a given γ in the cost function, both will have used the same Padé approximants for $\bar{w}_j(\beta)$. Consequently, they will both have the same misprediction on $\bar{w}_j(\beta)$. Leading to roughly the same misprediction on β_{opt}. However, in F_2 this misprediction on $\bar{w}_j(\beta)$ will get blown up because of raising the power, leading to a higher fractional difference.

The higher n, the more the $\beta_{\text{opt}}(\gamma)$ function approaches a constant function for $\gamma \in]0, 1[$, as explained in Section 2.2.2. This already makes it harder to approximate the real β_{opt} using the method we presented here (since we use continuous rational functions). For this class of cost functions, we should develop other techniques to achieve reasonable approximations. One can also wonder how relevant these kinds of cost functions with higher n are in practice.

For the first order approximation of $\beta_{\text{opt}}(\gamma)$, we can see a vertical jump in the graph which becomes more outspoken for higher n. A clear example hereof is the leftmost graph in Figure 4.6d at $\gamma = 0.7$. This is a consequence of the averaging over the set of $\phi(\beta)$ approximations, whereas the individual approximations in this set do not present this jump. We zoom in on the example of the first order approximation in Figure 4.6d. $\phi(\beta)$ approaches a step function with a very steep gradient at $\beta = 0.7$. Before averaging we have three clusters of approximations; each cluster approaches a step function with the step at another value for β. We see clusters with steps at β equal to 0.4, 0.65 and 0.72. Taking the average results in multiple steps at each of these values, where the height of the step is largely determined by the number of approximations in the cluster. Subsequently calculating the inverse, gives us (quasi-)horizontal parts in the graph at 0.4, 0.65 and 0.72 and the resulting jump at $\gamma = 0.7$ and $\gamma = 0.85$.

The other way to construct an approximation for $\beta_{\text{opt}}(\gamma)$ by using the average of the Padé approximants of $\bar{w}_j(\beta)$ in $\phi(\beta)$ and inverting that, is shown in Figure 4.7. We see that this alternative method does not present this jump. For the exotic case of $n = 20$ it even performs better, or at least more consistently bad. Where our proposed method performs worse for most γ it performs much better in a narrow range.

However, overall our tests have shown that this alternative method leads to less accurate results as is indicated in Figure 4.8. We see that the method proposed in Algorithm 4.2 performs about 3% better at the maximum misprediction point using the first order approximation. For the second order approximation this reduces to 0.5%.

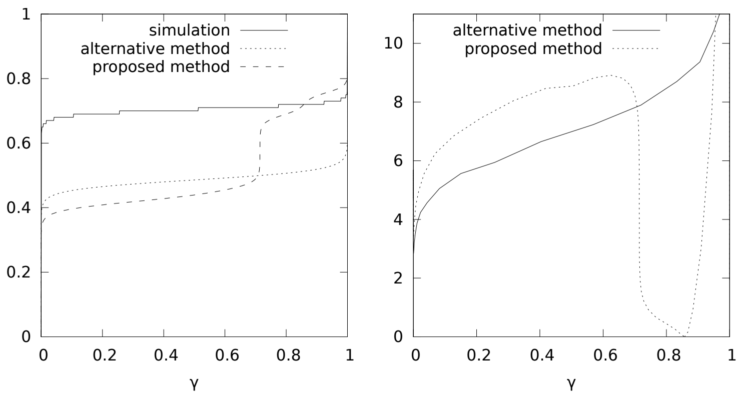
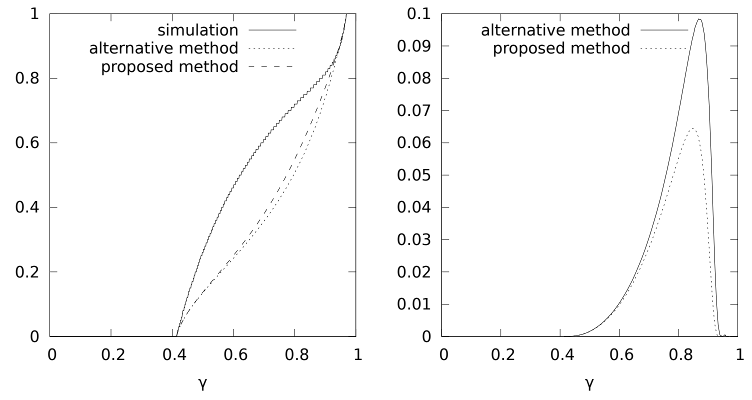


Figure 4.7: Comparison of first order approximations of two methods: $\beta_{\text{opt}}(\gamma)$ (left), fractional cost difference (right) for $n = 20$. The arrival process uses the following parameters: $N = 16$, $\alpha = 0.8$, $\lambda_T = 0.9$.

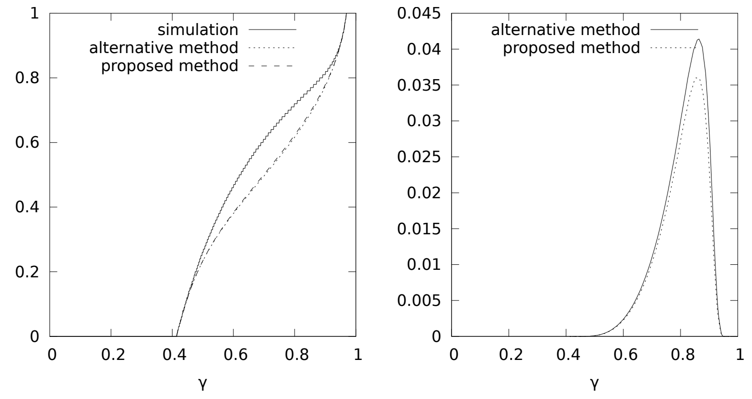
Lastly, for the third order approximation the difference between both methods further decreases, leading to quasi-identical results. For other parameters of the arrival process or other cost functions, we also see that both methods achieve the same results for the third order approximation. As this jump only presents itself when using cost functions that are not used in practice, we suggest to use the method we proposed in Algorithm 4.2. The exotic case for $n = 20$ is only mentioned here for completeness and a clearer illustration of this jump phenomenon.

4.5 The major challenge with the power series approximation

In the previous section, we constructed approximations based on truncated power series with maximum $M = 3$, i.e. with only 4 coefficients. This is because the actual calculation of the power series is far from straightforward. The reason for this is that l'Hôpital's rule has to be used multiple times, which leads to expressions for $V_m(z_1, z_2)$ that become more complex with m. Even worse, when we want to calculate the coefficients of the mean; the generating functions have to be differentiated in 1 which again requires multiple applications of l'Hôpital's rule. This was already mentioned in [145]. In this section, we elaborate on the problem and indicate some techniques to get the maximum out of the power series approximation.



(a) First order approximation



(b) Second order approximation

Figure 4.8: Comparison of two methods: $\beta_{\text{opt}}(\gamma)$ (left), fractional cost difference (right) for $n = 2$. The arrival process uses the following parameters: $N = 16$, $\alpha = 0.8$, $\lambda_T = 0.9$.

In the calculation of $Q_m(z_1, z_2)$ $(m > 1)$ using (4.6), $V_m(z_1, Y(z_1))$ and $V_m(0, z_2)$ each require the use of l'Hôpital's rule. Furthermore, for calculating the coefficients of the mean, the $V_m(z_1, z_2)$ functions need to be differentiated and evaluated in $z_1 = z_2 = 1$. This again requires the use of l'Hôpital's rule. As a result of all these differentiations and the recursive nature of the $V_m(z_1, z_2)$ expressions, the intermediate results in the calculations of the coefficients quickly become very large and hog up all computer memory. This made it impossible for us to calculate more than 4 coefficients in the power series for an unspecified generic arrival process.

To lessen the problem associated with l'Hôpital's rule, we did two modifications. The first one is to not first calculate $V_m(z_1, z_2)$ recursively as a function of only the arrival process. In concreto, to calculate $\mathrm{E}[w_1]_3$ we need the derivative of V_3 with respect to z_1 and evaluate it in $z_1 = z_2 = 1$. Therefore, we do this derivation and evaluation on the expression of $V_3(z_1, z_2)$ as a function of $V_2(z_1, z_2)$. This gives us an expression with certain derivatives of V_2; we calculate these subsequently as a function of V_1 and so on. The advantage is that these are all smaller expressions, which keeps the application of l'Hôpital's rule more tractable. The disadvantage is that we have lots of expressions, for instance, for $\mathrm{E}[w_1]_3$ we require $\left.\frac{\partial^3 V_0(z_1,z_2)}{\partial z_1^2 \partial z_2}\right|_{z_1=0, z_2=Y(0)}$ and 21 other expressions. Moreover, the amount of l'Hôpital's rule applications does not change and thus still rises quickly[5]. However, reuse of some of the intermediate expressions is possible when you need to calculate up to a certain coefficient, for instance, some expressions used in the calculation of $\mathrm{E}[w_1]_2$ can be reused for the calculation of $\mathrm{E}[w_1]_3$. To illustrate the problem, we show in Table 4.2 the number of evaluations of l'Hôpital's rule that we needed for the evaluation of each of the coefficients. This modification mainly makes it more tractable to implement an automated procedure, and (slightly) delays our l'Hôpital application limit.

The second modification relates to the evaluation. With our first modification, we now no longer have an analytical formula for $\mathrm{E}[w_1]_m$ as a function of the parameters of the arrival process only. We thus lose the possibility for an analytical sensitivity analysis. As such, the second modification is to directly use numerical values for the parameters of the arrival process. The result is that, the intermediate expressions are much smaller in memory as we now only have numerical coefficients for the V_m terms. Whereas if we only do the first modification we still have complicated (and increasingly larger in memory) expressions for the coefficients.

[5]On the order of m^4.

Coefficient	#l'Hôpital	Extra
$E[w_1]_0$	4	4
$E[w_1]_1$	5	5
$E[w_1]_2$	10	7
$E[w_1]_3$	48	40

Table 4.2: Required amount of applications of l'Hôpital's rule to obtain the desired coefficient (middle). Required extra amount of applications of l'Hôpital's rule to obtain up to the desired coefficient (right).

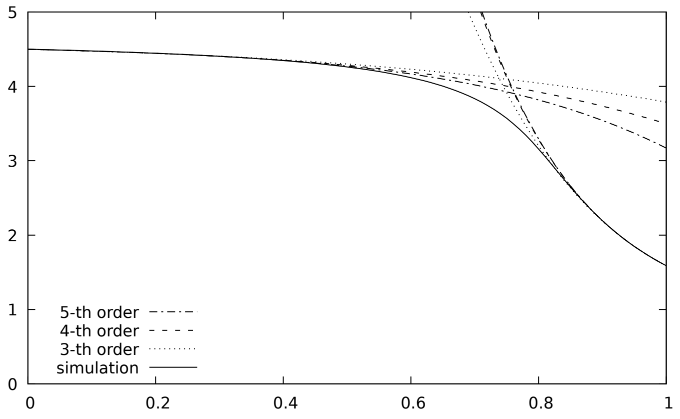


Figure 4.9: Truncated power series approximations for $\bar{w}_1(\beta)$ in $\beta = 0$ and $\beta = 1$. The arrival process is a two-dimensional binomial distribution with $N = 16, \lambda_T = 0.9, \alpha_1 = 0.8$.

Using these techniques, we implemented a procedure to automatically calculate the numerical coefficients for an arrival process with numerically specified parameters. With this procedure we were able to calculate 5 coefficients. In Figure 4.9, we show the power series resulting from these extra coefficients. In Figure 4.10, we show the average of the valid Padé approximants constructed using Algorithm 4.1. We see that extra coefficients indeed provide better approximations.

Undoubtedly, there are still some tweaks left to calculate even more coefficients, for instance, developing a better implementation in Maple, using a new Maple version or getting rid of Maple all together. A revolution in computer memory technology, so that more

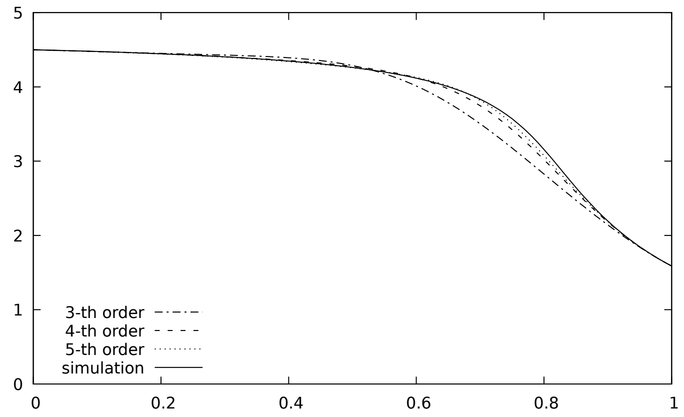


Figure 4.10: Average of valid Padé approximants for $\bar{w}_1(\beta)$ calculated from the truncated power series approximations in $\beta = 0$ and $\beta = 1$ using Algorithm 4.1. The arrival process is a two-dimensional binomial distribution with $N = 16, \lambda_T = 0.9, \alpha_1 = 0.8$.

memory is available might also provide improvements[6]. Clearly the main problem here is inherent to the method that implies the use of l'Hôpital's rule. In Chapter 6, we develop a new method to calculate a power series approximation of the mean unfinished work, without using generating functions and thus circumventing the problems associated with l'Hôpital's rule.

[6]We now used machines with 64GB RAM. Enabling a SWAP memory of 200GB on an SSD harddisk really put our patience to the test, but did not provide extra coefficients.

“Denk na, voor je iets doet.“
— Paul Pattyn

5 Three Classes

5.1 Introduction

In this chapter, the aim is to extend the power series approximation approach for two-class GPS queues (Section 4.2) to a three-class Hierarchical GPS queue as introduced in Section 3.1. This H-GPS model is illustrated in Figure 5.1. In each slot, assuming all queues are backlogged, the server first decides on the first hierarchical level to either serve queue 1 with probability β_1, or delegate service to hierarchical level 2 with probability $1-\beta_1$. If service is delegated to level 2, queue 2 is served with probability β_2 and queue 3 is served with probability $1-\beta_2$. If queue 1 is empty, service is by default delegated to level 2. Conversely, if queues 2 and 3 are empty queue 1 is served. If service is delegated to level 2 and one of the queues on that level is empty, the other queue is served. Furthermore, we assume that the arrival process is independent and identically distributed from slot to slot and that each customer requires a single slot of service.

This model with three classes leads to a random walk on the three-dimensional lattice in the positive octant, where the method of [41] is no longer applicable. Furthermore, to the best of our knowledge, analytical results for GPS models with three classes have not been obtained in the literature before.

We start our study for the special case $\beta_1 = 0$ in the next section. In this special case, classes 2 and 3 are served using GPS with pa-

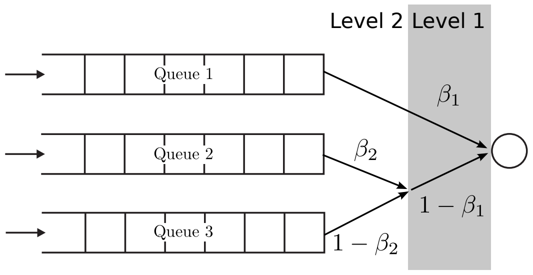


Figure 5.1: Discrete-time Hierarchical Generalized Processor Sharing

rameter β_2 and class 1 has lower priority than the other two classes. The dynamics of this model are quite simple: classes 2 and 3 are uninfluenced by class 1 and use two-class GPS for which we already have a power series approximation. Class 1 is just a regular lowest priority class. We use this special case as a stepping stone to the construction of a power series approximation centered around $\beta_1 = 0$ for the general model of Figure 5.1, in Section 5.3. Next, in Section 5.4, we study the special case for $\beta_1 = 1$. In this special case, class 1 has highest priority and the remaining capacity is shared using GPS among classes 2 and 3. Clearly, this special case is more complex than the special case $\beta_1 = 0$, as class 1 now influences the dynamics of the GPS-classes 2 and 3. As this is also a very interesting special case from a practical point of view, we conducted a thorough study of the influence of these high-priority customers on the remaining customers using numerical examples. With the special case $\beta_1 = 1$ in mind, we likewise construct a power series approximation centered around $\beta_1 = 1$ for the general model, in Section 5.5. Subsequently, in Section 5.6, we summarize the different power series for the general model that we constructed in this chapter. Lastly, in Section 5.7, we study some numerical examples that use the different power series.

5.2 GPS with an extra low-priority class

If we add an extra low-priority class to the two-class GPS model, we get the model in Figure 5.2. This corresponds to the special case $\beta_1 = 0$ in the three-class H-GPS model from Figure 5.1. In this case, queue 1 has low priority and is only served when both queues 2 and 3 are empty. In all other cases, the service of queues 2 and 3

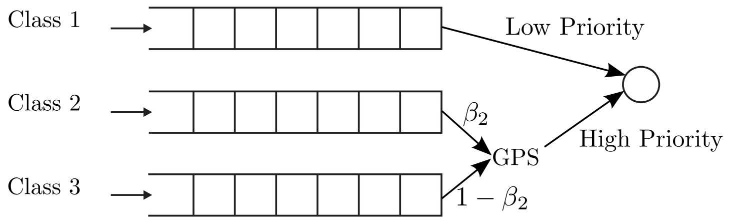


Figure 5.2: Special case: $\beta_1 = 0$.

is scheduled using two-class GPS with parameter β_2.

We define the joint pgf of the steady-state unfinished work probabilities of the three customer classes $W(z_1, z_2, z_3) = \mathrm{E}[z_1^{w_1} z_2^{w_2} z_3^{w_3}]$. Subsequently, we expand this pgf as a power series in β_2

$$W(z_1, z_2, z_3) = \sum_{m=0}^{\infty} V_m(z_1, z_2, z_3)\beta_2^m.$$

For $\beta_2 = 0$, we find $W(z_1, z_2, z_3)|_{\beta_2=0} = V_0(z_1, z_2, z_3)$. This pgf should correspond to that of the strict priority case $(3 > 2 > 1)$, which can be found from [144] and is written as:

$$V_0(z_1,z_2, z_3) \\ = \frac{(1-\lambda_T)A(z_1, z_2, z_3)(z_1-1)(z_2-Y_1(z_1))(z_3-Y_2(z_1, z_2))}{(z_1-Y_1(z_1))(z_2-Y_2(z_1, z_2))(z_3-A(z_1, z_2, z_3))},$$

whereby the functions Y_1 and Y_2 are implicitly defined as $Y_1(z_1) = A(z_1, Y_1(z_1), Y_1(z_1))$ and $Y_2(z_1, z_2) = A(z_1, z_2, Y_2(z_1, z_2))$. In the next section, we describe the origin and significance of these implicit functions in detail.

The coefficients $V_m(z_1, z_2, z_3)$ for $m > 0$ capture the GPS dynamics between classes 2 and 3. We can thus use the results from the power series for two-class GPS. We find for $m > 0$:

$$V_m(z_1, z_2, z_3) = \frac{A(z_1, z_2, z_3)(z_3-z_2)Q_{m-1}(z_1, z_2, z_3)}{z_2(z_3-A(z_1, z_2, z_3))},$$

where we define $Q_m(z_1, z_2, z_3) \triangleq V_m(z_1, z_2, z_3) - V_m(z_1, z_2, Y_2(z_1, z_2)) - V_m(z_1, 0, z_3) + V_m(z_1, 0, Y_2(z_1, z_2))$.

Analogously as for the two-class case, we can complement this power series centered around $\beta_2 = 0$ with a power series centered around $\beta_2 = 1$ by using the symmetry between classes 2 and 3. In concreto, $W(z_1, z_3, z_2) = \sum_{m=0}^{\infty} \tilde{V}_m(z_1, z_2, z_3)(1-\beta_2)^m$ whereby the expressions for $\tilde{V}_m$ are obtained by replacing $A(z_1, z_2, z_3)$ with $A(z_1, z_3, z_2)$, i.e. sending class-2 customers to queue 3 and vice versa. Note that in this expression z_3 (z_2) has now taken the role of the formal variable of the z-transform of the steady state unfinished work of class 2 (class 3), i.e. w_2 (w_3).

This special case could serve as a model for systems that share bandwidth among two traffic classes, while also offering a *best-effort* kind of service that just uses leftover bandwidth. This low-priority class could for instance be used for maintenance data, back-ups, etc. One specific example is as a model for cognitive radio which is explicitly designed to employ unused bandwidth [54, 81, 85, 113].

We skip the derivation of the mean values of the unfinished work in the queues in this section. The calculation of the mean value for class 2 or 3, is already possible from the results for the two-class system by ignoring class 1. The mean value for class 1 is calculated as the mean for a low-priority class by considering the aggregation of class 2 and 3 as a single high-priority class.

5.3 Power series approximation for H-GPS centered around $\beta_1 = 0$

5.3.1 Introducing the power series

In this section, we study the general model from Figure 5.1 for general β_1. In the previous section, we showed that a power series approximation for the special case $\beta_1 = 0$ is possible from the already obtained results. Now, we want to capture the influence of β_1 by constructing a power series centered around $\beta_1 = 0$. This construction is not that straightforward, that is why we start from the system equations. For ease of reference, we repeat them first.

- if $\boldsymbol{w}_k = \boldsymbol{0}$

$$\boldsymbol{w}_{k+1} = \boldsymbol{a}_k$$

- if $w_{i,k} > 0$; $w_{j,k} = 0$ for all $j \in \{1, 2, 3\} \setminus \{i\}$

$$\begin{aligned} w_{i,k+1} &= w_{i,k} - 1 + a_{i,k} \\ w_{j,k+1} &= a_{j,k} \end{aligned}$$

- if $w_{1,k} = 0$; $w_{2,k}, w_{3,k} > 0$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0, 1, 0) + \boldsymbol{a}_k \quad \text{w.p.} \quad \beta_2 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0, 0, 1) + \boldsymbol{a}_k \quad \text{w.p.} \quad 1 - \beta_2 \end{aligned}$$

- if $w_{2,k} = 0$; $w_{1,k}, w_{3,k} > 0$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (1,0,0) + \boldsymbol{a}_k && \text{w.p.} \quad \beta_1 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,0,1) + \boldsymbol{a}_k && \text{w.p.} \quad 1-\beta_1 \end{aligned}$$

- if $w_{3,k} = 0$; $w_{1,k}, w_{2,k} > 0$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (1,0,0) + \boldsymbol{a}_k && \text{w.p.} \quad \beta_1 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,1,0) + \boldsymbol{a}_k && \text{w.p.} \quad 1-\beta_1 \end{aligned}$$

- if $w_{j,k} > 0$ with $j = \{1,2,3\}$

$$\begin{aligned} \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (1,0,0) + \boldsymbol{a}_k && \text{w.p.} \quad \beta_1 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,1,0) + \boldsymbol{a}_k && \text{w.p.} \quad (1-\beta_1)\beta_2 \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - (0,0,1) + \boldsymbol{a}_k && \text{w.p.} \quad (1-\beta_1)(1-\beta_2) \end{aligned}$$

With $W_k(z_1, z_2, z_3) = \mathrm{E}[z_1^{w_{1,k}} z_2^{w_{2,k}} z_3^{w_{3,k}}]$ defined as the joint pgf of the unfinished work of the three classes, we transform the system equations to the following functional equation of this pgf:

$$W_{k+1}(z_1,z_2,z_3) = A(z_1,z_2,z_3) \cdot \tag{5.1}$$
$$\left(\begin{array}{l} W_k(0,0,0) + \frac{1}{z_1}\Big(W_k(z_1,0,0) - W_k(0,0,0)\Big) \\ + \frac{1}{z_2}\Big(W_k(0,z_2,0) - W_k(0,0,0)\Big) + \frac{1}{z_3}\Big(W_k(0,0,z_3) - W_k(0,0,0)\Big) \\ + \left(\frac{\beta_2}{z_2} + \frac{1-\beta_2}{z_3}\right)\Big(W_k(0,z_2,z_3) - W_k(0,0,z_3) - W_k(0,z_2,0) + W_k(0,0,0)\Big) \\ + \left(\frac{\beta_1}{z_1} + \frac{1-\beta_1}{z_2}\right)\Big(W_k(z_1,z_2,0) - W_k(z_1,0,0) - W_k(0,z_2,0) + W_k(0,0,0)\Big) \\ + \left(\frac{\beta_1}{z_1} + \frac{1-\beta_1}{z_3}\right)\Big(W_k(z_1,0,z_3) - W_k(z_1,0,0) - W_k(0,0,z_3) + W_k(0,0,0)\Big) \\ + \left(\frac{\beta_1}{z_1} + \frac{(1-\beta_1)\beta_2}{z_2} + \frac{(1-\beta_1)(1-\beta_2)}{z_3}\right) \cdot \\ \qquad \left(\begin{array}{l} W_k(z_1,z_2,z_3) - W_k(0,z_2,z_3) - W_k(z_1,0,z_3) - W_k(z_1,z_2,0) \\ +W_k(0,0,z_3) + W_k(0,z_2,0) + W_k(z_1,0,0) - W_k(0,0,0) \end{array} \right) \end{array} \right).$$

By letting $k \to \infty$ in (5.1) and with the stability condition $\lambda_T < 1$ fulfilled, we find a functional equation for the steady-state pgf $W(z_1,z_2,z_3) = \lim_{k\to\infty} W_k(z_1,z_2,z_3) = \lim_{k\to\infty} W_{k+1}(z_1,z_2,z_3)$. By rearranging terms, we find:

$$
\begin{aligned}
&W(z_1,z_2,z_3)\Big(z_1z_2z_3 - A(z_1,z_2,z_3)\begin{pmatrix} z_1z_2+\beta_1z_2(z_3-z_1) \\ +(1-\beta_1)\beta_2z_1(z_3-z_2)\end{pmatrix}\Big) \\
&\quad = A(z_1,z_2,z_3)\cdot \qquad\qquad (5.2)\\
&\quad\begin{pmatrix}
\beta_1\left(z_2(z_1-z_3)+\beta_2z_1(z_3-z_2)\right))W(0,z_2,z_3) \\
+(1-\beta_1)\beta_2z_1(z_2-z_3)W(z_1,0,z_3) \\
+(1-\beta_1)(1-\beta_2)z_1(z_3-z_2)W(z_1,z_2,0) \\
+(1-\beta_1)(z_3(z_2-z_1)+\beta_2z_1(z_3-z_2))W(z_1,0,0) \\
+\beta_1(1-\beta_2)z_1(z_3-z_2)W(0,z_2,0) \\
+\beta_1\beta_2z_1(z_2-z_3)W(0,0,z_3) \\
+\Big(z_2z_3(z_1-1)+\beta_1\beta_2z_1(z_3-z_2)+\beta_1z_3(z_2-z_1)\Big)W(0,0,0)
\end{pmatrix}.
\end{aligned}
$$

Instead of the three boundary functions for two-class GPS, we now need to eliminate seven boundary functions $W(0,z_2,z_3)$, $W(z_1,0,z_3)$, $W(z_1,z_2,0)$, $W(z_1,0,0)$, $W(0,z_2,0)$, $W(0,0,z_3)$, $W(0,0,0)$. We observe that $W(0,z_2,z_3)$, $W(0,z_2,0)$ and $W(0,0,z_3)$ only appear with coefficients with a factor β_1. Analogously as in the two-class case we therefore introduce a power series in β_1.

By assuming $W(z_1,z_2,z_3)$ is an analytic function of β_1 in a neighborhood of 0, we write $W(z_1,z_2,z_3)=\sum_{m=0}^{\infty}\breve{V}_m(z_1,z_2,z_3)\beta_1^m$ as its power series expansion in β_1 for all z_1, z_2 and z_3 in the complex unit disk. We substitute this power series in (5.2) and equate coefficients of corresponding powers of β_1. The result is the following functional equation for $\breve{V}_m$, for $m \geq 0$:

$$
\begin{aligned}
&\breve{V}_m(z_1,z_2,z_3)z_1\Big(z_2z_3 - A(z_1,z_2,z_3)\big(z_2-\beta_2(z_3-z_2)\big)\Big) \qquad (5.3)\\
&\quad + A(z_1,z_2,z_3)\begin{pmatrix}
z_1\beta_2(z_3-z_2)(\breve{V}_m(z_1,0,z_3)-\breve{V}_m(z_1,0,0)) \\
-z_1(1-\beta_2)(z_3-z_2)\breve{V}_m(z_1,z_2,0) \\
-z_3(z_2-z_1)\breve{V}_m(z_1,0,0) \\
-z_2z_3(z_1-1)\breve{V}_m(0,0,0)
\end{pmatrix} \\
&\quad = A(z_1,z_2,z_3)\begin{pmatrix}
z_1\beta_2(z_3-z_2)\begin{pmatrix}
-\breve{V}_{m-1}(z_1,z_2,z_3)+\breve{V}_{m-1}(0,z_2,z_3) \\
+\breve{V}_{m-1}(z_1,0,z_3)-\breve{V}_{m-1}(z_1,0,0) \\
-\breve{V}_{m-1}(0,0,z_3)+\breve{V}_{m-1}(0,0,0)
\end{pmatrix} \\
+z_1(1-\beta_2)(z_3-z_2)(\breve{V}_{m-1}(0,z_2,0)-\breve{V}_{m-1}(z_1,z_2,0)) \\
+z_2(z_3-z_1)(\breve{V}_{m-1}(z_1,z_2,z_3)-\breve{V}_{m-1}(0,z_2,z_3)) \\
+z_3(z_2-z_1)(\breve{V}_{m-1}(0,0,0)-\breve{V}_{m-1}(z_1,0,0))
\end{pmatrix},
\end{aligned}
$$

whereby we define $\breve{V}_{-1} \triangleq 0$. We assume that for a certain fixed m, $\breve{V}_{m-1}(z_1,z_2,z_3)$ is known and we want to find $\breve{V}_m(z_1,z_2,z_3)$. Then we have 4 unknowns left to eliminate: $\breve{V}_m(z_1,0,z_3)$, $\breve{V}_m(z_1,z_2,0)$, $\breve{V}_m(z_1,0,0)$ and $\breve{V}_m(0,0,0)$. Noting that some have a factor β_2 and others do not, we can reiterate our power series mechanism.

By assuming $\breve{V}_m(z_1, z_2, z_3)$ is an analytic function of β_2 in a neighborhood of 0, we write $\breve{V}_m(z_1, z_2, z_3) = \sum_{n=0}^{\infty} V_{m,n}(z_1, z_2, z_3)\beta_2^n$ as its power series expansion in β_2 for all z_1, z_2 and z_3 in the unit disk. Observe that we have actually expanded $W(z_1, z_2, z_3)$ as a two-dimensional power series $\sum_{n=0}^{\infty}\sum_{m=0}^{\infty} V_{m,n}(z_1, z_2, z_3)\beta_1^m\beta_2^n$ of (β_1, β_2) about the point $(0, 0)$. Subsequently, we substitute our power series for $\breve{V}_m$ in (5.3) and equate coefficients of corresponding powers of β_2. We then obtain a functional equation for $V_{m,n}$, for $m, n \geq 0$:

$$\begin{aligned}
&z_1z_2(z_3 - A(z_1, z_2, z_3))V_{m,n}(z_1, z_2, z_3) \qquad (5.4)\\
&\quad - z_1(z_3 - z_2)A(z_1, z_2, z_3)V_{m,n}(z_1, z_2, 0)\\
&\quad - z_3(z_2 - z_1)A(z_1, z_2, z_3)V_{m,n}(z_1, 0, 0)\\
&\quad - z_2z_3(z_1 - 1)A(z_1, z_2, z_3)V_{m,n}(0, 0, 0)\\
&=A(z_1, z_2, z_3)\cdot\\
&\left(\begin{array}{l}
z_2(z_3 - z_1)\Big(V_{m-1,n}(z_1, z_2, z_3) - V_{m-1,n}(0, z_2, z_3)\Big)\\
+z_1(z_3 - z_2)\left(\begin{array}{l}
V_{m,n-1}(z_1, z_2, z_3) - V_{m-1,n-1}(z_1, z_2, z_3)\\
+V_{m-1,n-1}(0, z_2, z_3) - V_{m,n-1}(z_1, 0, z_3)\\
+V_{m-1,n-1}(z_1, 0, z_3) - V_{m-1,n-1}(0, 0, z_3)\\
-V_{m,n-1}(z_1, z_2, 0) + V_{m-1,n-1}(z_1, z_2, 0)\\
+V_{m,n-1}(z_1, 0, 0) - V_{m-1,n-1}(z_1, 0, 0)\\
+V_{m-1,n}(0, z_2, 0) - V_{m-1,n-1}(0, z_2, 0)\\
-V_{m-1,n}(z_1, z_2, 0) + V_{m-1,n-1}(0, 0, 0)
\end{array}\right)\\
+z_3(z_2 - z_1)\Big(- V_{m-1,n}(z_1, 0, 0) + V_{m-1,n}(0, 0, 0)\Big)
\end{array}\right)\\
&\triangleq A(z_1, z_2, z_3)\left(\begin{array}{l}
z_2(z_3 - z_1)P^{(1)}_{m,n}(z_1, z_2, z_3)\\
+z_1(z_3 - z_2)P^{(2)}_{m,n}(z_1, z_2, z_3)\\
+z_3(z_2 - z_1)P^{(3)}_{m,n}(z_1)
\end{array}\right),
\end{aligned}$$

whereby we define $V_{m,n} = 0$ for m or $n < 0$ and in the last line we define $P^{(1)}_{m,n}(z_1, z_2, z_3)$, $P^{(2)}_{m,n}(z_1, z_2, z_3)$ and $P^{(3)}_{m,n}(z_1)$ as short for the expression in brackets in the three terms on the previous line. Assuming for fixed m, n $V_{m',n'}(z_1, z_2, z_3)$ with $m'+n' < m+n$ is known, then we are left with determining $V_{m,n}(z_1, z_2, 0)$, $V_{m,n}(z_1, 0, 0)$ and $V_{m,n}(0, 0, 0)$ to fully specify $V_{m,n}(z_1, z_2, z_3)$.

By using Rouché's theorem (see next section), we find that $z_3 - A(z_1, z_2, z_3)$ has one zero in the unit disk of z_3 for arbitrary $|z_1| < 1$ and $|z_2| < 1$. We denote this zero by $Y_2(z_1, z_2)$, which is thus defined by the implicit definition: $Y_2(z_1, z_2) = A(z_1, z_2, Y_2(z_1, z_2))$. In fact, as for the two-class case, we will again show that this is a joint pgf of random variables. Setting $z_3 = Y_2(z_1, z_2)$ in (5.4), we find

$$\begin{aligned}
&-z_1(Y_2(z_1,z_2)-z_2)V_{m,n}(z_1,z_2,0) \\
&-Y_2(z_1,z_2)(z_2-z_1)V_{m,n}(z_1,0,0) \\
&-z_2Y_2(z_1,z_2)(z_1-1)V_{m,n}(0,0,0) \\
&\quad= \begin{pmatrix} z_2(Y_2(z_1,z_2)-z_1)P^{(1)}_{m,n}(z_1,z_2,Y_2(z_1,z_2)) \\ +z_1(Y_2(z_1,z_2)-z_2)P^{(2)}_{m,n}(z_1,z_2,Y_2(z_1,z_2)) \\ +Y_2(z_1,z_2)(z_2-z_1)P^{(3)}_{m,n}(z_1) \end{pmatrix}.
\end{aligned} \tag{5.5}$$

Once again using Rouché's theorem, we find that $Y_2(z_1, z_2) - z_2$ has one zero in the unit disk of z_2 for arbitrary $|z_1| < 1$. We denote this zero by $Y_1(z_1)$; it is defined in the unit disk as $Y_1(z_1) = Y_2(z_1, Y_1(z_1))$ and $|Y_1(z_1)| < 1$. Setting $z_2 = Y_1(z_1)$ in (5.5), we find

$$\begin{aligned}
&-(Y_1(z_1)-z_1)V_{m,n}(z_1,0,0) - Y_1(z_1)(z_1-1)V_{m,n}(0,0,0) \\
&\quad= (Y_1(z_1)-z_1)(P^{(1)}_{m,n}(z_1,Y_1(z_1),Y_1(z_1)) + P^{(3)}_{m,n}(z_1)).
\end{aligned} \tag{5.6}$$

Lastly, we argue that $W(0,0,0)$ equals $\Pr[w_T = w_1 + w_2 + w_3 = 0]$, i.e., the probability that the complete system is empty, which we know is only related to the arrival process and not to the scheduling (since it is work-conserving). As such, $W(0,0,0)$ equals $1 - \lambda_T$ and does not depend on β_1 or β_2. Since $W(0,0,0) = \sum_{n=0}^{\infty}\sum_{m=0}^{\infty} V_{m,n}(0,0,0)$ $\beta_1^m\beta_2^n$, we find that $V_{0,0}(0,0,0) = 1 - \lambda_T$ and $V_{m,n}(0,0,0)$ equals 0, for all other cases.

We can now recursively work our way back through the expressions to finally obtain a fully specified expression for $V_{m,n}(z_1, z_2, z_3)$. Using $V_{m,n}(0,0,0)$ in (5.6) we find $(m + n > 0)$:

$$\begin{aligned}
V_{m,n}(z_1,0,0) &= -P^{(1)}_{m,n}(z_1,Y_1(z_1),Y_1(z_1)) - P^{(3)}_{m,n}(z_1) \\
V_{0,0}(z_1,0,0) &= \frac{Y_1(z_1)(z_1-1)(1-\lambda_T)}{z_1-Y_1(z_1)},
\end{aligned} \tag{5.7}$$

where we used that $P^{(i)}_{0,0} = 0$ $(i = 1, \ldots, 3)$. Using (5.7) in (5.5):

$$\begin{aligned}
V_{m,n}(z_1,z_2,0) =& \frac{\begin{pmatrix} z_2(Y_2(z_1,z_2)-z_1)P^{(1)}_{m,n}(z_1,z_2,Y_2(z_1,z_2)) \\ -Y_2(z_1,z_2)(z_2-z_1)P^{(1)}_{m,n}(z_1,Y_1(z_1),Y_1(z_1)) \end{pmatrix}}{z_1(z_2-Y_2(z_1,z_2))} \\
&- P^{(2)}_{m,n}(z_1,z_2,Y_2(z_1,z_2)) \\
V_{0,0}(z_1,z_2,0) =& \frac{(1-\lambda_T)(z_1-1)Y_2(z_1,z_2)(z_2-Y_1(z_1))}{(z_2-Y_2(z_1,z_2))(z_1-Y_1(z_1))}.
\end{aligned} \tag{5.8}$$

Summary of power series expansion of $W(z_1, z_2, z_3)$ in $(\beta_1, \beta_2) = (0, 0)$

For $W(z_1, z_2, z_3) \triangleq \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} V_{m,n}(z_1, z_2, z_3)\beta_1^m \beta_2^n$, we find:

$$V_{0,0}(z_1, z_2, z_3) = \frac{(1-\lambda_T)A(z_1, z_2, z_3)(z_1 - 1)(z_2 - Y_1(z_1))(z_3 - Y_2(z_1, z_2))}{(z_1 - Y_1(z_1))(z_2 - Y_2(z_1, z_2))(z_3 - A(z_1, z_2, z_3))},$$

with implicitly defined $Y_2(z_1, z_2) = A(z_1, z_2, Y_2(z_1, z_2))$ and $Y_1(z_1) = Y_2(z_1, Y_1(z_1))$. Furthermore, for $m + n > 0$ we find:

$$\begin{aligned} V_{m,n}(z_1, z_2, z_3) =& \frac{A(z_1, z_2, z_3)(z_3 - z_2)Q_{m,n-1}(z_1, z_2, z_3)}{z_2(z_3 - A(z_1, z_2, z_3))} \\ &+ \frac{A(z_1, z_2, z_3)(z_3 - z_1)(V_{m-1,n}(z_1, z_2, z_3) - V_{m-1,n}(0, z_2, z_3))}{z_1(z_3 - A(z_1, z_2, z_3))} \\ &+ \frac{A(z_1, z_2, z_3)(z_2 - z_3)(z_1 - Y_2(z_1, z_2))(V_{m-1,n}(z_1, z_2, Y_2(z_1, z_2)) - V_{m-1,n}(0, z_2, Y_2(z_1, z_2)))}{z_1(z_2 - Y_2(z_1, z_2))(z_3 - A(z_1, z_2, z_3))} \\ &+ \frac{A(z_1, z_2, z_3)(z_1 - z_2)(z_3 - Y_2(z_1, z_2))(V_{m-1,n}(z_1, Y_1(z_1), Y_1(z_1)) - V_{m-1,n}(0, Y_1(z_1), Y_1(z_1)))}{z_1(z_2 - Y_2(z_1, z_2))(z_3 - A(z_1, z_2, z_3))}, \end{aligned} \tag{5.9}$$

with

$$\begin{aligned} Q_{m,n-1}(z_1, z_2, z_3) =& P_{m,n}^{(2)}(z_1, z_2, z_3) - P_{m,n}^{(2)}(z_1, z_2, Y_2(z_1, z_2)) \\ =& V_{m,n-1}(z_1, z_2, z_3) - V_{m,n-1}(z_1, z_2, Y_2(z_1, z_2)) - V_{m-1,n-1}(z_1, z_2, z_3) + V_{m-1,n-1}(z_1, z_2, Y_2(z_1, z_2)) \\ &+ V_{m-1,n-1}(0, z_2, z_3) - V_{m-1,n-1}(0, z_2, Y_2(z_1, z_2)) - V_{m,n-1}(z_1, 0, z_3) + V_{m,n-1}(z_1, 0, Y_2(z_1, z_2)) \\ &+ V_{m-1,n-1}(z_1, 0, z_3) - V_{m-1,n-1}(z_1, 0, Y_2(z_1, z_2)) - V_{m-1,n-1}(0, 0, z_3) + V_{m-1,n-1}(0, 0, Y_2(z_1, z_2)). \end{aligned}$$

And finally with (5.8) in (5.4), we find the expressions of the power series that are summarized on page 141. Note that it is only because of significant simplifications that all terms have a common index ($n-1$ for Q), which we also make explicit in the new definition. This also helps in better seeing the structure of the equations.

In the next subsection, we prove that the application of Rouché's theorem to obtain Y_2 and Y_1 is justified. Furthermore, we show that they represent stochastic variables and as such explain their function in the equations. In the subsection thereafter, we study the structure of the equations. We will show the correctness of the equations and improve the intuition about what they represent. In the last subsection, we look at some numerical examples, to show their use and performance in practice.

5.3.2 The implicitly defined functions Y_1 and Y_2

In the previous section, we introduced Y_1 and Y_2 as unique zeros of respectively $Y_2(z_1, z_2) - z_2$ and $z_3 - A(z_1, z_2, z_3)$ and stated that the existence of this zeros could be proven using Rouché's theorem. In this section, we actually prove this statement and show that each function is the pgf of a stochastic variable. We start by stating Rouché's theorem for completeness.

Theorem 5.1. *Let $f(z)$ and $g(z)$ be two analytic functions inside and on a closed contour C in the complex z-plane such that $|g(z)| < |f(z)|$ for all z on C. Then the functions $f(z)$ and $f(z)+g(z)$ have the same number of zeros inside C.*

First we look at $Y_2(z_1, z_2)$. For each $|z_1| < 1$, $|z_2| < 1$, we define $f(z_3) = z_3$ and $g(z_3) = A(z_1, z_2, z_3)$. We then show that $|g(z_3)| < |f(z_3)|$ for all $|z_3| = 1$:

$$\begin{aligned}
|g(z_3)| &= \left| \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} \Pr[a_1 = i, a_2 = j, a_3 = k] z_1^i z_2^j z_3^k \right| \\
&\leq \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} \Pr[a_1 = i, a_2 = j, a_3 = k] |z_1|^i |z_2|^j |z_3|^k \\
&= \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} \Pr[a_1 = i, a_2 = j, a_3 = k] |z_1|^i |z_2|^j \\
&< 1 = |f(z_3)|,
\end{aligned}$$

where we have consecutively used the triangle inequality, the fact that $|z_3| = 1$ and, in the last step, $|z_1| < 1, |z_2| < 1$ and the normalization condition. According to Rouché's theorem, $z_3 - A(z_1, z_2, z_3)$ then has exactly one zero in the unit disk for arbitrary $|z_1| < 1$ and $|z_2| < 1$. We denote this zero by $Y_2(z_1, z_2)$ and it is thus defined by the equation $Y_2(z_1, z_2) - A(z_1, z_2, Y_2(z_1, z_2)) = 0$ or equivalently by $Y_2(z_1, z_2) = A(z_1, z_2, Y_2(z_1, z_2))$. The introduction of $Y_2(z_1, z_2)$ in the previous section was thus justified.

Furthermore, we show that $Y_2(z_1, z_2)$ is the joint pgf of the number of class-1 and class-2 arrivals during a sub-busy period[1] initiated by a random arrival in the queueing system with priority for class-3 and where classes 1 and 2 are served in their order of arrival. Define $y_{2,1}$ as the number of class-1 arrivals and $y_{2,2}$ as the number of class-2 arrivals in such sub-busy period. We then write, analogously as in Section 4.2, the numbers of arrivals in a tagged sub-busy period as:

$$\begin{aligned}
y_{2,1} =& a_1^{(1)} + \sum_{l=1}^{a_3^{(1)}} y_{2,1,l}^{(1)} \\
y_{2,2} =& a_2^{(1)} + \sum_{l=1}^{a_3^{(1)}} y_{2,2,l}^{(1)},
\end{aligned}$$

whereby $a_j^{(1)}$ denotes the class-j arrivals in the first slot of the tagged sub-busy period and $y_{2,j,l}^{(1)}$ the number of class-j arrivals in the sub-busy period initiated by the l-th class-3 arrival in the first slot of the tagged sub-busy period. Taking a simple z-transform we find:

$$\begin{aligned}
Y_2(z_1, z_2) =& \mathrm{E}\left[z_1^{y_{2,1}} z_2^{y_{2,2}}\right] \\
=& \mathrm{E}\left[z_1^{a_1^{(1)}} z_2^{a_2^{(1)}} z_1^{\sum_{l=1}^{a_3^{(1)}} y_{2,1,l}^{(1)}} z_2^{\sum_{l=1}^{a_3^{(1)}} y_{2,2,l}^{(1)}}\right] \\
=& \mathrm{E}\left[z_1^{a_1^{(1)}} z_2^{a_2^{(1)}} \left(z_1^{y_{2,1}} z_2^{y_{2,2}}\right)^{a_3^{(1)}}\right] \\
=& A(z_1, z_2, Y_2(z_1, z_2)),
\end{aligned}$$

by noting in the second step that $y_{2,j}$ and $y_{2,j,l}^{(1)}$ have an identical distribution and are independent.

Next we prove the existence of the zero $Y_1(z_1)$ of $Y_2(z_1, z_2) - z_2$. For each $|z_1| < 1$, we define $f(z_2) = z_2$ and $g(z_2) = Y_2(z_1, z_2)$. We

[1]We defined the notion of a sub-busy period in the sidenote in Section 4.2.

show that $|g(z_2)| < |f(z_2)|$ for all $|z_2| = 1$:

$$\begin{aligned}|g(z_2)| =& \left|\sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\Pr[y_1 = i, y_2 = j] z_1^i z_2^j\right| \\ \leq& \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\Pr[y_1 = i, y_2 = j]|z_1|^i|z_2|^j \\ =& \sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\Pr[y_1 = i, y_2 = j]|z_1|^i \\ <& 1 = |f(z_2)|,\end{aligned}$$

where we again consecutively used the triangle inequality, the fact that $|z_2| = 1$ and in the last step $|z_1| < 1$. According to Rouché's theorem $Y_2(z_1, z_2) - z_2$ then has exactly one zero in the unit disk for arbitrary $|z_1| < 1$ and normalization. We denote this zero by $Y_1(z_1)$ and it is thus defined by the equation $Y_2(z_1, Y_1(z_1)) - Y_1(z_1) = 0$ or equivalently by $Y_1(z_1) = Y_2(z_1, Y_1(z_1))$. The introduction of $Y_1(z_1)$ in the previous section was thus justified.

Using the implicit definition of Y_2 in the implicit definition of $Y_1(z_1) = Y_2(z_1, Y_1(z_1))$, we find $Y_1(z_1) = A(z_1, Y_1(z_1), Y_2(z_1, Y_1(z_1)))$ which in turn equals $Y_1(z_1) = A(z_1, Y_1(z_1), Y_1(z_1))$. We prove that $Y_1(z_1)$ is the pgf of the number of class-1 arrivals during a sub-busy period initiated by a random arrival in the strict priority system obtained by setting $\beta_1 = \beta_2 = 0$, i.e. $(3 > 2 > 1)$[2]. Defining y_1 as the number of class-1 arrivals in such a sub-busy period, we then write:

$$y_1 = a_1^{(1)} + \sum_{l=1}^{a_2^{(1)}+a_3^{(1)}} y_{1,l}^{(1)}.$$

The z-transform, subsequently yields,

$$\begin{aligned}Y_1(z_1) =& \mathrm{E}\left[z_1^{y_1}\right] \\ =& \mathrm{E}\left[z_1^{a_1^{(1)}} (z_1^{y_1})^{a_2^{(1)}} (z_1^{y_1})^{a_3^{(1)}}\right] \\ =& A(z_1, Y_1(z_1), Y_1(z_1)).\end{aligned}$$

[2] Note that any priority system where classes 2 and 3 have priority over class 1 suffices. For instance, the system $(2 > 3 > 1)$, i.e. $\beta_1 = 0, \beta_2 = 1$, yields an equivalent $Y_1(z_1)$. In fact, $Y_1(z_1)$ is identical to the implicit function from a two-class priority system with low priority for class 1 and high priority for the aggregation of classes 2 and 3.

Equivalently, we can also use the stochastic variables $y_{2,1}$ and $y_{2,2}$ from Y_2 to define y_1:

$$y_1 = y_{2,1} + \sum_{l=1}^{y_{2,2}} y_{1,l},$$

because any class-2 arrival ($y_{2,2}$) now also has priority over class-1 and initiates new sub-busy periods with arrival opportunities for class-1. Z-transforming the previous expression gives

$$\begin{aligned} Y_1(z_1) =& \mathrm{E}\left[z_1^{y_1}\right] \\ =& \mathrm{E}\left[z_1^{y_{2,1}}(z_1^{y_1})^{y_{2,2}}\right] \\ =& Y_2(z_1, Y_1(z_1)), \end{aligned}$$

which is indeed equal to our first implicit function definition of $Y_1(z_1)$.

Lastly, we note that the implicit functions $Y_2(z_1, z_2)$ and $Y_1(z_1)$ are pgf's of numbers of arrivals in sub-busy periods in a related priority queueing system. The functions we used here are equivalent to the implicit functions defined in [144] for multi-class strict priority queueing systems. Moreover, they are identical to the implicit functions that were used in Section 5.2. Furthermore, y_1 is equal to the number of class-1 arrivals in a two-queue system whereby we label our original class-2 and class-3 customers with the same label. $Y_1(z_1)$ here is thus equivalent to $Y(z_1)$ of Section 4.2 given that we aggregate class-2 and class-3 customers in queue 2 of that system. This can also be seen from the implicit function definition $Y_1(z_1) = A(z_1, Y_1(z_1), Y_1(z_1))$.

5.3.3 Seeing the patterns

In this section, we look at some already known special cases of our H-GPS system. This way we test the correctness of our solution and also get a better insight in the patterns that govern the expressions we obtained. A first test is setting $\beta_1 = \beta_2 = 0$, which corresponds to the priority queueing system $(3 > 2 > 1)$. For that system, we find $W(z_1, z_2, z_3)|_{\beta_1=\beta_2=0} = V_{0,0}(z_1, z_2, z_3)$ which corresponds with the results of [144]. If, we aggregate classes 2 and 3 in one high-priority class, we obtain a two-queue priority system $(3, 2 > 1)$. So $\mathrm{E}[z_1^{w_1} z_{23}^{w_2+w_3} | \beta_1 = \beta_2 = 0] = W(z_1, z_{23}, z_{23})|_{\beta_1=\beta_2=0} = V_{0,0}(z_1, z_{23}, z_{23})$,

$$V_{0,0}(z_1, z_{23}, z_{23}) = \frac{(1-\lambda_T)A(z_1, z_{23}, z_{23})(z_1 - 1)(z_{23} - Y_1(z_1))}{(z_1 - Y_1(z_1))(z_{23} - A(z_1, z_{23}, z_{23}))}.$$

This is indeed the pgf of the unfinished work in a two-class priority queue [143]. Note that, we also find this last result for any other value β_2.

If we only aggregate classes 2 and 3 (for general β_1 and β_2), we obtain a two-class GPS system. Checking this with our expressions, we find $\mathrm{E}[z_1^{w_1} z_{23}^{w_2+w_3}] = W(z_1, z_{23}, z_{23}) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} V_{m,n}(z_1, z_{23}, z_{23})$ $\beta_1^m \beta_2^n$ and

$$\begin{aligned}
&V_{m,n}(z_1, z_{23}, z_{23})\\
&\quad= \frac{A(z_1, z_{23}, z_{23})(z_{23} - z_1)}{z_1(z_{23} - A(z_1, z_{23}, z_{23}))} \cdot \\
&\quad\left(\begin{array}{l} V_{m-1,n}(z_1, z_{23}, z_{23}) - V_{m-1,n}(0, z_{23}, z_{23}) \\ -V_{m-1,n}(z_1, Y_1(z_1), Y_1(z_1)) + V_{m-1,n}(0, Y_1(z_1), Y_1(z_1)) \end{array} \right)
\end{aligned}$$

We note that $V_{m,n}(z_1, z_{23}, z_{23})$ with $n > 0$ is recursively defined (via $V_{m-1,n}$) in terms of $V_{-1,n}(z_1, z_{23}, z_{23})$ which equals 0. With $V_{m,n}(z_1, z_{23}, z_{23}) = 0$ for $n > 0$, $W(z_1, z_{23}, z_{23})$ simplifies to the single dimensional series $\sum_{m=0}^{\infty} V_{m,0}(z_1, z_{23}, z_{23})\beta_1^m$, which indeed eliminates the superfluous β_2 and corresponds with the power series approximation we found in Section 4.2.

By setting $\beta_1 = 0$ and ignoring class-1 customers, we should also find a two-queue GPS system with classes 2 and 3. We write: $\mathrm{E}[z_2^{w_2} z_3^{w_3} | \beta_1 = 0] = W(1, z_2, z_3)|_{\beta_1=0} = \sum_{n=0}^{\infty} V_{0,n}(1, z_2, z_3)\beta_2^n$.

$$\begin{aligned}
V_{0,n}(1, z_2, z_3) =& \frac{A(1, z_2, z_3)(z_3 - z_2)Q_{0,n-1}(1, z_2, z_3)}{z_2(z_3 - A(1, z_2, z_3))} \\
V_{0,0}(1, z_2, z_3) =& \frac{1 - \lambda_T}{1 - Y_1'(1)} \frac{A(1, z_2, z_3)(z_2 - 1)(z_3 - Y_2(1, z_2))}{(z_2 - Y_2(1, z_2))(z_3 - A(1, z_2, z_3))},
\end{aligned}$$

whereby we used l'Hôpital's rule in the last line. To find $Y_1'(1)$ we derive the implicit definition $Y_1(z_1) = A(z_1, Y_1(z_1), Y_1(z_1))$ to find

$$Y_1'(1) = \frac{\lambda_1}{1 - \lambda_2 - \lambda_3},$$

and thus

$$V_{0,0}(1, z_2, z_3) = \frac{(1 - \lambda_2 - \lambda_3)A(1, z_2, z_3)(z_2 - 1)(z_3 - Y_2(1, z_2))}{(z_2 - Y_2(1, z_2))(z_3 - A(1, z_2, z_3))}.$$

Furthermore

$$\begin{aligned}
Q_{0,n-1}(1, z_2, z_3) =& V_{0,n-1}(1, z_2, z_3) - V_{0,n-1}(1, z_2, Y_2(1, z_2)) \\
&- V_{0,n-1}(1, 0, z_3) + V_{0,n-1}(1, 0, Y_2(1, z_2)).
\end{aligned}$$

We thus indeed find the expected expressions for the power series of a two-class system. Note how λ_T got replaced by the total load of the two considered classes via Y_1'.

As a last example, for $\beta_1 = 0$, we find $W(z_1, z_2, z_3)|_{\beta_1=0} = \sum_{n=0}^{\infty} V_{0,n}(z_1, z_2, z_3)\beta_2^n$. Using (5.9) for the calculation of these coefficients, we obtain:

$$\begin{aligned}
V_{0,n}(z_1, z_2, z_3) =& \frac{A(z_1, z_2, z_3)(z_3 - z_2)Q_{0,n-1}(z_1, z_2, z_3)}{z_2(z_3 - A(z_1, z_2, z_3))}, \\
Q_{0,n-1}(z_1, z_2, z_3) =& V_{0,n-1}(z_1, z_2, z_3) - V_{0,n-1}(z_1, z_2, Y_2(z_1, z_2)) \\
& - V_{0,n-1}(z_1, 0, z_3) + V_{0,n-1}(z_1, 0, Y_2(z_1, z_2)).
\end{aligned}$$

These expressions indeed correspond to the expressions from the special case in Section 5.2.

In this section, we have shown that the power series for three-class H-GPS includes all previous obtained results for the relevant configurations. This verifies the correctness of the result and strengthens intuition on the construction of the expression. In the next section, we use the power series approximation to approximate the performance measures and compare them with simulation results.

5.3.4 Numerical examples

In Section 5.3.1, we developed a power series for the joint pgf of the unfinished work around the point $(\beta_1, \beta_2) = (0, 0)$. Analogously, as in the two-class case we derive a power series from this for the mean unfinished work of the classes. Note that we concentrate on the mean unfinished work of class 2 in the remainder. The mean unfinished work of class 1 is, because of the hierarchical nature of the system, independent of β_2 and can be calculated from the two-class system by aggregating classes 2 and 3 in one new class. Because of the symmetry in the system, the expressions for the mean unfinished work of class 3 are analogous to those for class 2.

$$\bar{w}_2 = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \left.\frac{\partial V_{m,n}(z_1, z_2, z_3)}{\partial z_2}\right|_{z_1=z_2=z_3=1} \beta_1^m \beta_2^n$$

This is a power series centered around $(\beta_1, \beta_2) = (0, 0)$, we can thus expect it to have deteriorating accuracy for larger β_1 or β_2.

Like in the two-class system, we can use the symmetry between classes 2 and 3 to construct a power series around $(\beta_1, \beta_2) = (0, 1)$:

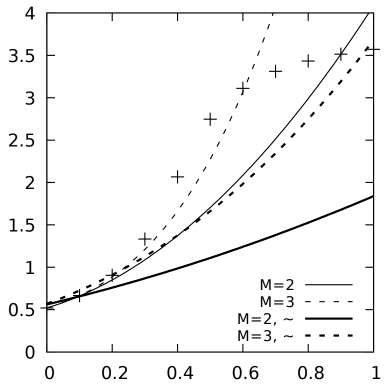
$$\bar{w}_2 = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \left.\frac{\partial \tilde{V}_{m,n}(z_1, z_2, z_3)}{\partial z_3}\right|_{z_1=z_2=z_3=1} \beta_1^m (1 - \beta_2)^n,$$

whereby $\tilde{V}_{m,n}$ has $A(z_1, z_2, z_3)$ replaced with $A(z_1, z_3, z_2)$. This means we send class-2 customers to queue 3 and calculate the unfinished work in that queue (derivative w.r.t. z_3).

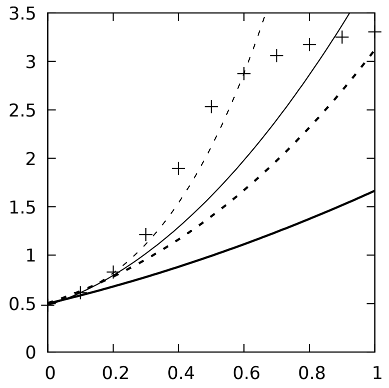
In Figures 5.3 and 5.4, we study the performance of the approximation. We used a multinomial distribution (see Section 1.6) with $N = 16, \lambda_T = 0.9, \alpha_1 \triangleq \frac{\lambda_1}{\lambda_T} = 0.4, \alpha_2 \triangleq \frac{\lambda_2}{\lambda_T} = 0.3$ as an arrival process. Each of the graphs displays the truncated power series of $\bar{w}_2$, whereby M denotes the degree of the polynomial. The $\sim$ is used to denote the polynomial that is calculated using the symmetric system and is drawn in thicker lines. The $+$-symbols denote simulation results. In Figure 5.3 we show $\bar{w}_2$ as a function of β_1 for fixed values of $\beta_2 = \{0, 0.2, 0.8, 1\}$. We see in the top left graph ($\beta_2 = 0$) that the regular approximation is perfect for $\beta_1 = 0$. In the neighborhood of $\beta_1 = 0$ the approximation is good and improves by adding extra terms in the power series. For higher β_2, we see that the accuracy of the approximation steadily declines. On the other hand the approximation by the symmetric systems improves. In the bottom right graph for $\beta_2 = 0$, the approximation matches the derivatives perfectly. These observations are expected, by the very definition of the power series.

In Figure 5.4 $\bar{w}_2$ is displayed as a function of β_2 for fixed values of $\beta_1 = \{0, 0.2, 0.8, 1\}$. In the top left graph, we see that for $\beta_2 = 0$ the regular power series performs well as does the symmetric power series for $\beta_2 = 1$, as expected. For $\beta_1 = 0.2$ (top right graph), the accuracy of both is reasonable. However for $\beta_1 = 0.8$ or 1 the approximations are practically unusable.

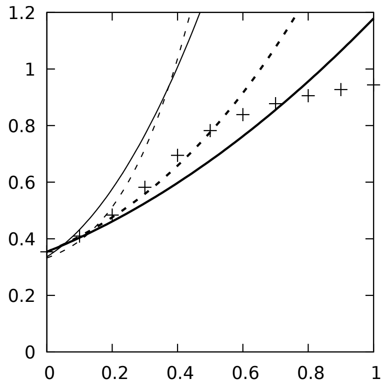
In general, as expected, the truncated power series approximation deteriorates as we move further away from the expansion points of the power series, i.e., the points $(\beta_1, \beta_2) = (0, 0)$ and $(\beta_1, \beta_2) = (0, 1)$. To derive a power series centered around $(\beta_1, \beta_2) = (1, 0)$ or $(\beta_1, \beta_2) = (1, 1)$, we cannot exploit symmetry of the system. If we were to send the arrivals of class 2 or 3 to queue 1 the whole system dynamic changes. We will come back to this in Section 5.5, where we develop a power series approximation around these aforementioned expansion points. As a stepping stone, in the next section, we first study the special case where $\beta_1 = 1$, in that way creating a priority class above two GPS classes.



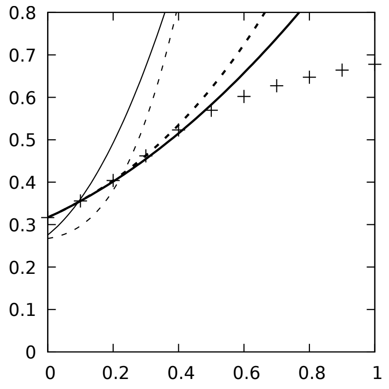
(a) $\bar{w}_2(\beta_1, 0)$



(b) $\bar{w}_2(\beta_1, 0.2)$

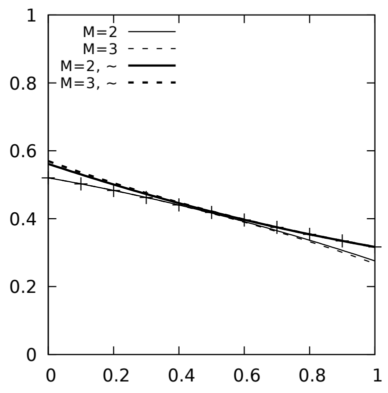


(c) $\bar{w}_2(\beta_1, 0.8)$

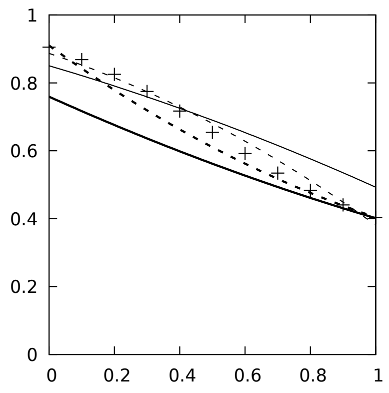


(d) $\bar{w}_2(\beta_1, 1)$

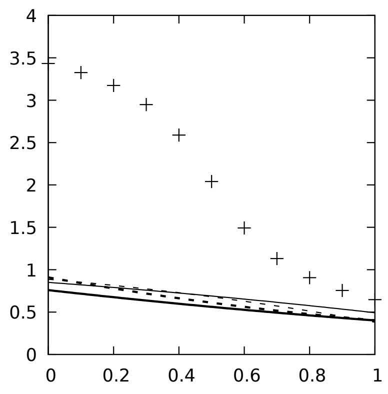
Figure 5.3: Truncated power series for $\bar{w}_2(\beta_1, \beta_2)$ with $N = 16$, $\lambda_T = 0.9$, $\alpha_1 = 0.4$, $\alpha_2 = 0.3$. The $\sim$ indicates the cases where we used the symmetric system.



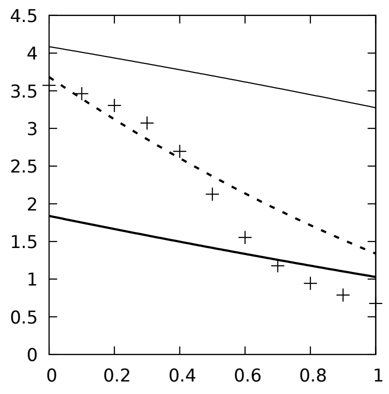
(a) $\bar{w}_2(0, \beta_2)$



(b) $\bar{w}_2(0.2, \beta_2)$



(c) $\bar{w}_2(0.8, \beta_2)$



(d) $\bar{w}_2(1, \beta_2)$

Figure 5.4: Truncated power series for $\bar{w}_2(\beta_1, \beta_2)$ with $N = 16$, $\lambda_T = 0.9$, $\alpha_1 = 0.4$, $\alpha_2 = 0.3$. The $\sim$ indicates the cases where we used the symmetric system.

5.4 Influence of high-priority customers on a GPS queue

Numerous queueing systems in practice, have a high-priority bypass possibility. For instance, the processor of a computer system is shared by several jobs, whereby each class of jobs gets a time-share according to the weight of its class. However, the processor can also be interrupted, for hardware I/O for instance (i.e., the user pushes a key, requested data from the harddisk becomes available ...), these are in fact short high-priority jobs, bypassing the normal scheduling mechanism.

An example from telecommunications is DiffServ [108]. DiffServ is short for Differentiated Services and is an architecture designed to deliver a different Quality of Service (QoS) grade to various services in telecommunication networks. It defines an Expedited Forwarding (EF) class of packets next to the Assured Forwarding (AF) class. EF packets have essentially high priority and are thus given strict priority over all other packets. The AF class of packets is divided into subclasses, and the scheduling amongst the subclasses is a GPS-based scheduling.

Cisco implemented this kind of scheduling mechanism in some of its gigabit switch routers. The brand names used are IP Real-time Transport Protocol (RTP) Priority and Low Latency Queueing (LLQ); both are based on a mixture of GPS-like scheduling with priority bypassing [36]. They differ in the type of traffic they support, i.e., UDP vs TCP.

As a result of its practical application, this model also attracted attention from the research community, where it is frequently referred to as PQ-GPS. Jin et al. [79, 80] studied PQ-GPS under long-range dependent traffic by using a flow decomposition approach dividing the system into single-server single-queue (SSSQ) systems. They obtain analytical upper and lower bounds. Parveen [111] used the same SSSQ approach to study a system containing both long-range and short-range dependent traffic. After the single queue decomposition he however uses another technique resulting in a single approximation, as opposed to an upper and lower bound. Lastly, we mention Wang et al. [148] who studied a finite hybrid queueing model using PQ and Weighted Fair Queueing (WFQ). As WFQ is known to be a good approximation for GPS, it is also of interest here. Drawing up a Markov chain for the system and solving it for the steady-state probability, they conclude with a sensitivity analysis for the parameters of the system.

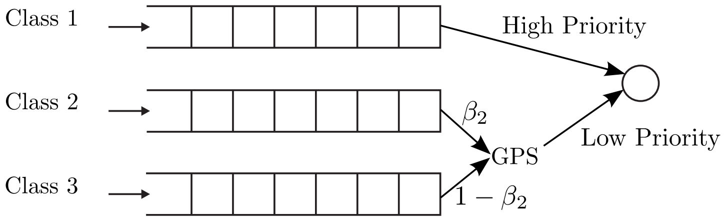


Figure 5.5: Model PQ-GPS

In this section, we study the influence of these high-priority customers on a generalized processor sharing (GPS) queue. This model corresponds with a H-GPS model with $\beta_1 = 1$, thus giving class-1 high priority while letting classes 2 and 3 share the remaining slots with GPS. As such it is a special case of our general model in Section 5.3.1. However, as we have seen in the numerical results, the performance of the power series approximation for the general model performs bad for β_1 values not in the neighborhood of 0. The solution is to repeat the method we used there and apply it to this specific setting. We summarize the model in Figure 5.5.

5.4.1 The power series approximation

We can follow the exact same steps as in the derivation of the power series for the general model (Section 5.3.1). We start with setting $\beta_1 = 1$ in the functional equation (5.2) to obtain the functional equation for this case:

$$\begin{aligned} &W(z_1,z_2,z_3)z_2z_3\Big(z_1 - A(z_1,z_2,z_3)\Big) \\ &=A(z_1,z_2,z_3)\cdot \qquad\qquad (5.10)\\ &\left(\begin{array}{l} (z_2(z_1-z_3)+\beta_2 z_1(z_3-z_2)))\,W(0,z_2,z_3) \\ +(1-\beta_2)z_1(z_3-z_2)W(0,z_2,0) \\ +\beta_2 z_1(z_2-z_3)W(0,0,z_3) \\ +\Big(z_1z_3(z_2-1)+\beta_2 z_1(z_3-z_2)\Big)W(0,0,0) \end{array}\right). \end{aligned}$$

Note that this functional equation only includes boundary functions with $z_1 = 0$, these include the system dynamics in the important situation when $w_1 = 0$, i.e. no high-priority class-1 customers are present in the system. Subsequently, we introduce the power series $W(z_1,z_2,z_3) = \sum_{m=0}^{\infty} V_m(z_1,z_2,z_3)\beta_2^m$ in (5.10). Equating corresponding coefficients of β_2, we find:

$$\begin{aligned} &z_2z_3\Big(z_1 - A(z_1,z_2,z_3)\Big)V_m(z_1,z_2,z_3) \\ &\quad - z_2(z_1-z_3)A(z_1,z_2,z_3)V_m(0,z_2,z_3) \\ &\quad - z_1(z_3-z_2)A(z_1,z_2,z_3)V_m(0,z_2,0) \\ &\quad - z_1z_3(z_2-1)A(z_1,z_2,z_3)V_m(0,0,0) \\ &= A(z_1,z_2,z_3)z_1(z_3-z_2)P_{m-1}(z_2,z_3), \end{aligned} \tag{5.11}$$

whereby for short, we defined $P_m(z_2,z_3) \triangleq V_m(0,z_2,z_3) - V_m(0,z_2,0) - V_m(0,0,z_3) + V_m(0,0,0)$ and $V_{-1}(z_1,z_2,z_3) \triangleq 0$.

By using Rouché's theorem, analogous as in Section 5.3.2, we show that $z_1 - A(z_1,z_2,z_3)$ has exactly one zero in the unit disk of z_1 for arbitrary $|z_2| < 1$ and $|z_3| < 1$. We denote this zero by $Y_2(z_2,z_3)$ and it is thus implicitly defined by $Y_2(z_2,z_3) = A(Y_2(z_2,z_3),z_2,z_3)$. Furthermore, $Y_2(z_2,z_3)$ is the joint pgf of the number of class-2 and class-3 arrivals during a sub-busy period initiated by a random arrival in the queueing system with priority for class-1 and where classes 2 and 3 are served in their order of arrival[3]. Using $z_1 = Y_2(z_2,z_3)$ in (5.11), we obtain

$$\begin{aligned} &-z_2(Y_2(z_2,z_3)-z_3)V_m(0,z_2,z_3) \\ &\quad - Y_2(z_2,z_3)(z_3-z_2)V_m(0,z_2,0) \\ &\quad - Y_2(z_2,z_3)z_3(z_2-1)V_m(0,0,0) \\ &= Y_2(z_2,z_3)(z_3-z_2)P_{m-1}(z_2,z_3). \end{aligned} \tag{5.12}$$

Once more, we use Rouché's theorem and show that $Y_2(z_2,z_3) - z_3$ has one zero in the unit disk of z_3 for arbitrary $|z_2| < 1$. We denote it by $Y_1(z_2)$ and it is implicitly defined by $Y_1(z_2) = Y_2(z_2,Y_1(z_2)) = A(Y_1(z_2),z_2,Y_1(z_2))$. $Y_1(z_2)$ is the pgf of the number of class-2 arrivals during a sub-busy period initiated by a random arrival in the queueing system with priority for classes 1 and 3. Substituting $z_3 = Y_1(z_2)$ in (5.12), gives

$$\begin{aligned} &-(Y_1(z_2)-z_2)V_m(0,z_2,0) \\ &\quad - Y_1(z_2)(z_2-1)V_m(0,0,0) \\ &= (Y_1(z_2)-z_2)P_{m-1}(z_2,Y_1(z_2)). \end{aligned} \tag{5.13}$$

Using $W(0,0,0) = \sum_{m=0}^{\infty} V_m(0,0,0)\beta_2^m = 1 - \lambda_T$, we find that $V_0(0,0,0) = 1 - \lambda_T$ and $V_m(0,0,0) = 0$ for $m > 0$. Subsequently, we

[3] Note that Rouché's theorem could already be used on (5.10) to introduce Y_2. However, we did not do that, for symmetry reasons in the methodology. Obviously, the end result is identical.

can work our way back up to $V_m(z_1, z_2, z_3)$. We start with (5.13) and find for $m > 0$:

$$\begin{aligned} V_0(0, z_2, 0) =& \frac{Y_1(z_2)(z_2 - 1)(1 - \lambda_T)}{z_2 - Y_1(z_2)} \\ V_m(0, z_2, 0) =& - P_{m-1}(z_2, Y_1(z_2)). \end{aligned}$$

We substitute these expressions in (5.12) to obtain

$$\begin{aligned} V_0(0, z_2, z_3) =& \frac{Y_2(z_2, z_3)(1 - \lambda_T)(z_2 - 1)(z_3 - Y_1(z_2))}{(z_2 - Y_1(z_2))(z_3 - Y_2(z_2, z_3))} \\ V_m(0, z_2, z_3) =& \frac{Y_2(z_2, z_3)(z_3 - z_2)\Big(P_{m-1}(z_2, z_3) - P_{m-1}(z_2, Y_1(z_2))\Big)}{z_2(z_3 - Y_2(z_2, z_3))}. \end{aligned}$$

Lastly, we feed the expressions for $V_m(0, z_2, 0)$ and $V_m(0, z_2, z_3)$ back into (5.11) and find the expressions that make up our recursive definition of the power series:

$$\begin{aligned} &V_0(z_1, z_2, z_3) \\ &\quad = \frac{A(z_1, z_2, z_3)(1 - \lambda_T)(z_2 - 1)(z_1 - Y_2(z_2, z_3))(z_3 - Y_1(z_2))}{(z_1 - A(z_1, z_2, z_3))(z_2 - Y_1(z_2))(z_3 - Y_2(z_2, z_3))} \\ &V_m(z_1, z_2, z_3) = \frac{A(z_1, z_2, z_3)(z_3 - z_2)(z_1 - Y_2(z_2, z_3))Q_{m-1}(z_2, z_3)}{z_2(z_1 - A(z_1, z_2, z_3))(z_3 - Y_2(z_2, z_3))}, \end{aligned}$$

whereby we have defined

$$\begin{aligned} Q_m(z_2, z_3) \triangleq& P_m(z_2, z_3) - P_m(z_2, Y_1(z_2)) \\ =& V_m(0, z_2, z_3) - V_m(0, z_2, Y_1(z_2)) \\ & - V_m(0, 0, z_3) + V_m(0, 0, Y_1(z_2)). \end{aligned} \tag{5.14}$$

We note that with $z_2 = z_3 = z_{23}$, which corresponds to aggregating queues 2 and 3, $V_m(z_1, z_{23}, z_{23}) = 0$ for $m > 0$. This is what we expect as $W(z_1, z_{23}, z_{23})$ is independent of β_2 (the total unfinished work in the low-priority queues is independent of β_2 due to the work-conserving nature of GPS).

5.4.2 Approximations of performance measures

As for the previous power series of pgf's, we now study how to derive expressions for the mean unfinished work. As before, we write

$$\begin{aligned} \bar{w}_j =& \left. \frac{\partial W(z_1, z_2, z_3)}{\partial z_j} \right|_{z_1, z_2, z_3 = 1} \\ =& \sum_{m=0}^{\infty} \beta_2^m \left. \frac{\partial V_m(z_1, z_2, z_3)}{\partial z_j} \right|_{z_1, z_2, z_3 = 1}. \end{aligned}$$

Considering that $V_m(z_1, 1, 1) = 0$ for $m > 0$ we find

$$\bar{w}_1 = \left. \frac{\partial V_0(z_1, 1, 1)}{\partial z_1} \right|_{z_1=1},$$

the mean unfinished work in queue 1 indeed does not depend on β_2. Furthermore as $\bar{w}_T = \bar{w}_1 + \bar{w}_2 + \bar{w}_3$ is also independent of β_2, we find for $m > 0$ (analogously to the two-class case)

$$\left. \frac{\partial V_m(1, z_2, 1)}{\partial z_2} \right|_{z_2=1} = - \left. \frac{\partial V_m(1, 1, z_3)}{\partial z_3} \right|_{z_3=1}.$$

This eliminates the need to calculate the coefficients for the power series of both $\bar{w}_2$ and $\bar{w}_3$ separately.

In addition, we can calculate the power series around $\beta_2 = 1$ by exploiting the symmetry of the model. As before, we therefore send class-2 customers to queue 3 and vice versa.

$$\bar{w}_2 = \sum_{m=0}^{\infty} \left. \frac{\partial \tilde{V}_m(1, 1, z_3)}{\partial z_3} \right|_{z_3=1} (1 - \beta_2)^m,$$

whereby $\tilde{V}_m$ is defined by replacing $A(z_1, z_2, z_3)$ by $A(z_1, z_3, z_2)$ in V_m.

With truncated power series in both $\beta_2 = 0$ and $\beta_2 = 1$, we can set the Padé approximants to work again. Furthermore as on the first level we have invariable strict priority, we know $\bar{w}_2$ should decrease with increasing β_2. We could thus employ an adapted version of Algorithm 4.1. However, we will just use the pure approximants for the numerical examples in the next section.

5.4.3 Numerical examples

In this section, we compare our power series approximation for the mean unfinished work with simulation results. As the unfinished work for class 1 is not influenced by the other queues and could also easily be calculated from results for single-class FCFS queueing, we will not discuss it here. Furthermore, we only analyze queue 2, as the results for queue 3 follow easily from the work conserving properties of the system. We have used a multinomial arrival process with $N = 16$.

In Figure 5.6, we show the mean unfinished work of class 2 as a function of the weight β_2, with $\lambda_T = 0.9, \alpha_1 = 0.1$, and $\alpha_2 = 0.1$. The figure shows curves of the simulation result and the Padé approximants without poles. We can see that for these parameters the [2/3] Padé approximant is very accurate. Secondly, we reconfirm that

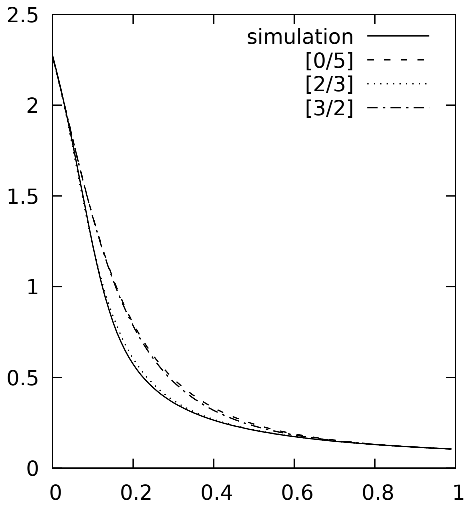


Figure 5.6: $\bar{w}_2(\beta_2)$: comparison between simulation and Padé approximants.

the approximations perform best close to $\beta_2 = 0$ and $\beta_2 = 1$. This is expected as the available information is exactly the value up to the M-th order derivative in these points (in this case $M = 2$). Subsequently, the approximants are constructed to match this information, thus performing well near $\beta_2 = 0$ and $\beta_2 = 1$.

In a second numerical example, we study the influence of the amount of high-priority (i.e., class-1) customers. We keep the total load $\lambda_T = 0.9$ fixed and $\lambda_2 = \lambda_3$, while increasing α_1 from 0.1 to 0.6. $\bar{w}_2(\beta_2)$ is depicted in Figure 5.7 on the left, showing both the simulation results and the best performing Padé approximant. We can see that the performance of the approximation is still accurate though slightly deteriorates as α_1 decreases, this results from the choice of the approximant. For this graph, we chose the [3/2] approximant, which on average performs best for these curves, but for smaller α_1 the [2/3] approximant is actually better. Furthermore for $\beta_2 = 1$, i.e., when the queueing system is effectively a strict priority system with class 1 having highest priority, class 2 medium priority and class 3 low priority, higher α_1 barely makes a difference. This is mainly because there are few class-2 customers in the system as α_2 decreases from 0.45 to 0.2. On the other end for $\beta_2 = 0$, we have a strict priority queueing system with class 1 high priority, class 3 medium priority

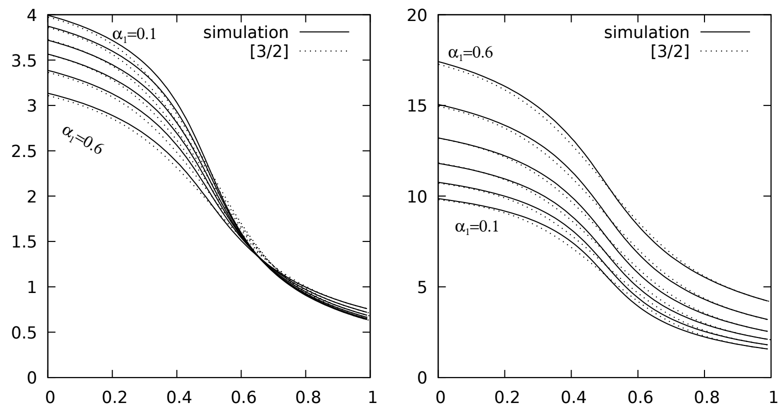


Figure 5.7: Mean unfinished work of class 2 (left) and mean class-2 delay (right) as a function of β_2: effect of increasing fraction of class-1 customers

and class 2 low priority. As class 2 is the lowest on the priority ladder, the influence of bypassing (higher priority) class-3 and class-1 customers is greater.

Using Little's law, we also calculated the mean class-2 delay; it is depicted in Figure 5.7 on the right. We saw before that as α_1 increases the mean queue-2 unfinished work decreases, mainly because α_2 decreases (we keep the total load and ratio between class-2 and 3 packets fixed). As we can see from the graph of the delay, for an increasing amount of high-priority packets the class-2 packets have a larger delay. There are thus less class-2 packets in the system but they stay there longer.

In Figure 5.8, we show $\bar{w}_2$ as a function of β_2 for different values of the total load λ_T, with $\alpha_1 = \alpha_2 = 0.1$ fixed. As the load in the system increases, we observe the queue-2 unfinished work increases as well. This is a classical queueing result: a higher load always leads to higher congestion. As in the previous example (and for the same reason), we can see the effect at $\beta_2 = 1$ is barely visible as opposed to at $\beta_2 = 0$. Furthermore, we see that the approximation is close to the simulated result. For $\lambda_T = 0.99$, we see that because of the high load the approximation deteriorates, especially for small β_2.

Lastly, we look at the influence of the amount of class-2 customers while keeping the total load and the amount of high-priority packets constant. The results are depicted in Figure 5.9 for $\lambda_T = 0.9, \alpha_1 = 0.1$ and α_2 ranging from 0.1 to 0.5. As the amount of class-2 packets

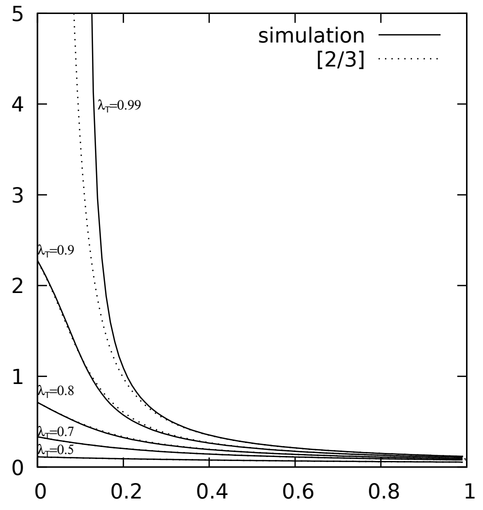


Figure 5.8: Mean class-2 unfinished work as a function of β_2: effect of increasing total load

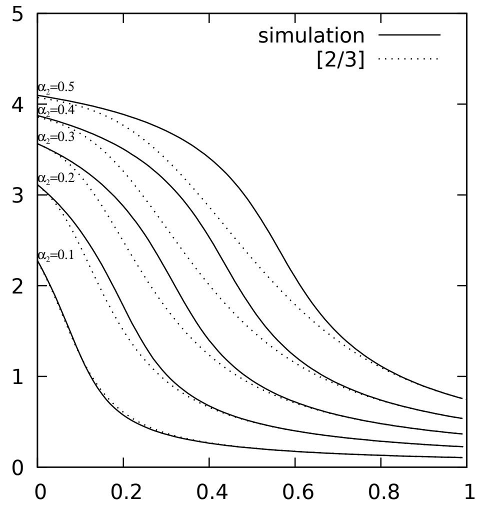


Figure 5.9: Mean class-2 unfinished work as a function of β_2: effect of increasing fraction of class-2 customers

increases the unfinished work increases, which was to be expected. Another observation is that the performance of the approximation deteriorates. In Figure 5.9, we chose to show the [2/3] approximant. This is, however, not the best approximation for every parameter combination. For instance, for $\alpha_2 = 0.5$ Padé approximant [3/2] is the best one. However, even if we compare every simulation with the best fitting approximant, the performance still deteriorates.

For the numerical results in this section, we have generally used the best fitting Padé approximant. In practice, however, it is impossible to know which Padé approximant has the best fit, because the availability of simulations results would largely defeat the purpose of constructing the approximation in the first place. A solution to the selection problem of the best approximant is an extension of Algorithm 4.1, which we discussed in length in the previous chapter.

5.5 Power series approximation for H-GPS centered around $\beta_1 = 1$

In Section 5.3, we developed a power series approximation for the joint pgf of the unfinished work in the queues of the three-class H-GPS system. This power series was centered around the point $(\beta_1, \beta_2) = (0, 0)$. Via the symmetric system, i.e. interchanging classes 2 and 3, we also obtained a power series about the point $(\beta_1, \beta_2) = (0, 1)$. As expected, we established that the performance of the approximation deteriorates for points (β_1, β_2) further away from these expansion points. In the previous section, we studied the H-GPS system with $\beta_1 = 1$, therefor constructing a power series approximation about $\beta_2 = 0$ or 1. Note that this was a one-dimensional power series (parameter β_2). However, it showed that the system with $\beta_1 = 1$ can be studied using our power series approach. In this section, we construct a two-dimensional power series for the general H-GPS system, whereby the series is centered around $(\beta_1, \beta_2) = (1, 0)$. Via the symmetric system, we then also obtain a power series centered around $(\beta_1, \beta_2) = (1, 1)$.

We introduce the power series about $(\beta_1, \beta_2) = (1, 0)$:

$$W(z_1, z_2, z_3) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \hat{V}_{m,n}(z_1, z_2, z_3)(1 - \beta_1)^m \beta_2^n. \tag{5.15}$$

In the next step, we perform the substitution of (5.15) in the functional equation of the H-GPS system (5.2). Subsequently, as before, we equate corresponding coefficients of powers of β_1 and β_2, resulting in a functional equation for the coefficients of the power series $\hat{V}_{m,n}$. The remaining steps are analogous as in Section 5.3.1. Using Rouché's theorem twice, we introduce the implicitly defined functions $\hat{Y}_2(z_2, z_3) = A(\hat{Y}_2(z_2, z_3), z_2, z_3)$ and $\hat{Y}_1(z_2) = \hat{Y}_2(z_2, \hat{Y}_1(z_2)) = A(\hat{Y}_1(z_2), z_2, \hat{Y}_1(z_2))$. Note that these functions are identical[4] to the ones defined in the previous section for the special case $\beta_1 = 1$. Ultimately, we obtain the expressions on pages 161-162.

Note that for $\beta_1 = 1$ the coefficients $\hat{V}_{0,n}(z_1, z_2, z_3)$ are the only ones remaining in the power series. For $n > 0$, we see that only the last term in the expression for $\hat{V}_{0,n}$ remains and only the first four in $\hat{Q}^{(2)}_{0,n-1}$. The remaining terms and expressions indeed correspond to the power series we found in Section 5.4.1. As a second check, setting $z_2 = z_3 = z_{23}$, we indeed find the expression for the symmetric two-class GPS power series approximation.

Using the symmetry between classes 2 and 3 in the H-GPS system, we derive a power series about $(\beta_1, \beta_2) = (1, 1)$. We write:

$$W(z_1, z_3, z_2) = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} \hat{\hat{V}}_{m,n}(z_1, z_2, z_3)(1 - \beta_1)^m (1 - \beta_2)^n,$$

whereby $\hat{\hat{V}}_{m,n}$ has $A(z_1, z_2, z_3)$ replaced with $A(z_1, z_3, z_2)$, i.e. we send class-2 customers to queue 3 and vice versa. Note that in this expression z_3 (z_2) has now taken the role of the formal variable of the z-transform of the steady state unfinished work of class 2 (class 3), i.e. w_2 (w_3).

5.6 Summary of power series for H-GPS

Collecting all results from this chapter, we summarize the four different power series for E$[w_2]$, whereby the superscript denotes the

[4] We use the $\hat{}$-symbol here to differentiate $\hat{Y}_1$ and $\hat{Y}_2$ from Y_1 and Y_2 used in the power series about $\beta_1 = 0$. Likewise, we also use $\hat{V}_{m,n}$ to differentiate from $V_{m,n}$.

Summary of power series expansion of $W(z_1, z_2, z_3)$ in $(\beta_1, \beta_2) = (1, 0)$

For $W(z_1, z_2, z_3) \triangleq \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} \hat{V}_{m,n}(z_1, z_2, z_3)(1-\beta_1)^m \beta_2^n$, we find:

$$\hat{V}_{0,0}(z_1, z_2, z_3) = \frac{(1-\lambda_T)A(z_1, z_2, z_3)(z_2 - 1)(z_1 - \hat{Y}_2(z_2, z_3))(z_3 - \hat{Y}_1(z_2))}{(z_1 - A(z_1, z_2, z_3))(z_2 - \hat{Y}_1(z_2))(z_3 - \hat{Y}_2(z_2, z_3))},$$

with implicitly defined

$$\begin{aligned}\hat{Y}_1(z_2) &= A(\hat{Y}_1(z_2), z_2, \hat{Y}_1(z_2))\\ \hat{Y}_2(z_2, z_3) &= A(\hat{Y}_2(z_2, z_3), z_2, z_3).\end{aligned}$$

Furthermore, we find for $m + n > 0$:

$$\begin{aligned}\hat{V}_{m,n}(z_1, z_2, z_3) =& \frac{A(z_1, z_2, z_3)(z_1 - z_3)\Big(\hat{V}_{m-1,n}(z_1, z_2, z_3) - \hat{V}_{m-1,n}(\hat{Y}_2(z_2, z_3), z_2, z_3)\Big)}{z_3(z_1 - A(z_1, z_2, z_3))}\\ &+ \frac{A(z_1, z_2, z_3)(z_2 - z_1)\hat{V}_{m-1,n}(z_1, 0, 0)}{z_2(z_1 - A(z_1, z_2, z_3))}\\ &+ \frac{A(z_1, z_2, z_3)(z_1 - z_3)(z_2 - \hat{Y}_2(z_2, z_3))\hat{V}_{m-1,n}(\hat{Y}_2(z_2, z_3), 0, 0)}{z_2(z_1 - A(z_1, z_2, z_3))(z_3 - \hat{Y}_2(z_2, z_3))}\\ &+ \frac{A(z_1, z_2, z_3)(z_3 - z_2)(z_1 - \hat{Y}_2(z_2, z_3))\hat{V}_{m-1,n}(\hat{Y}_1(z_2), 0, 0)}{z_2(z_1 - A(z_1, z_2, z_3))(z_3 - \hat{Y}_2(z_2, z_3))}\end{aligned}$$

$$+\frac{A(z_1,z_2,z_3)(z_3-z_2)\begin{pmatrix} z_1(z_3-\hat{Y}_2(z_2,z_3))\hat{Q}^{(1)}_{m-1,n}(z_1,z_2,z_3) \\ -z_3(z_1-\hat{Y}_2(z_2,z_3))\hat{Q}^{(1)}_{m-1,n}(\hat{Y}_1(z_2),z_2,\hat{Y}_1(z_2)) \\ +\hat{Y}_2(z_2,z_3)(z_1-z_3)\hat{Q}^{(1)}_{m-1,n}(\hat{Y}_2(z_2,z_3),z_2,z_3) \end{pmatrix}}{z_2z_3(z_1-A(z_1,z_2,z_3))}$$
$$+\frac{A(z_1,z_2,z_3)(z_3-z_2)(z_1-\hat{Y}_2(z_2,z_3))\hat{Q}^{(2)}_{m,n-1}(z_1,z_2,z_3)}{z_2(z_1-A(z_1,z_2,z_3))(z_3-\hat{Y}_2(z_2,z_3))},$$

with

$$\begin{aligned}
\hat{Q}^{(1)}_{m-1,n}(z_1,z_2,z_3) &\triangleq \hat{V}_{m-1,n-1}(z_1,z_2,z_3)-\hat{V}_{m-1,n-1}(z_1,0,z_3)-\hat{V}_{m-1,n-1}(z_1,z_2,0)+\hat{V}_{m-1,n}(z_1,z_2,0)+\hat{V}_{m-1,n-1}(z_1,0,0)\\
\hat{Q}^{(2)}_{m,n-1}(z_1,z_2,z_3) &\triangleq \hat{V}_{m,n-1}(0,z_2,z_3)-\hat{V}_{m,n-1}(0,z_2,\hat{Y}_1(z_2))-\hat{V}_{m,n-1}(0,0,z_3)+\hat{V}_{m,n-1}(0,0,\hat{Y}_1(z_2))\\
&\quad -\hat{V}_{m-1,n-1}(0,z_2,z_3)+\hat{V}_{m-1,n-1}(0,z_2,\hat{Y}_1(z_2))+\hat{V}_{m-1,n-1}(0,0,z_3)-\hat{V}_{m-1,n-1}(0,0,\hat{Y}_1(z_2)).
\end{aligned}$$

expansion point,

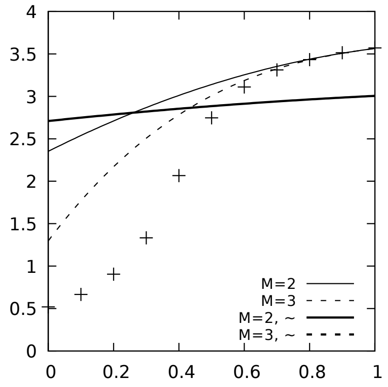
$$
\begin{aligned}
\mathrm{E}[w_2]^{(0,0)} &= \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \left.\frac{\partial V_{m,n}(z_1,z_2,z_3)}{\partial z_2}\right|_{z_1=z_2=z_3=1} \beta_1^m \beta_2^n \\
\mathrm{E}[w_2]^{(0,1)} &= \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \left.\frac{\partial \tilde{V}_{m,n}(z_1,z_2,z_3)}{\partial z_3}\right|_{z_1=z_2=z_3=1} \beta_1^m (1-\beta_2)^n \\
\mathrm{E}[w_2]^{(1,0)} &= \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \left.\frac{\partial \hat{V}_{m,n}(z_1,z_2,z_3)}{\partial z_2}\right|_{z_1=z_2=z_3=1} (1-\beta_1)^m \beta_2^n \\
\mathrm{E}[w_2]^{(1,1)} &= \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \left.\frac{\partial \hat{\tilde{V}}_{m,n}(z_1,z_2,z_3)}{\partial z_3}\right|_{z_1=z_2=z_3=1} (1-\beta_1)^m (1-\beta_2)^n.
\end{aligned}
$$

Note that it is also possible to calculate expressions for power series expanded about $(\beta_1, \beta_2) = (0, 1)$ and $(\beta_1, \beta_2) = (1, 1)$ using the exact same method as for $(\beta_1, \beta_2) = (0, 0)$ and $(\beta_1, \beta_2) = (1, 0)$. That way the symmetry of the system is not exploited, see also the discussion on two-class GPS in Chapter 4. Furthermore, if a computer algorithm is implemented to calculate the power series about $(\beta_1, \beta_2) = (0, 0)$, the algorithm can be reused for the calculation of the power series about $(\beta_1, \beta_2) = (0, 1)$, using the symmetry of the system. By not exploiting the symmetry, one would need to implement a new algorithm from the power series about $(\beta_1, \beta_2) = (0, 1)$.

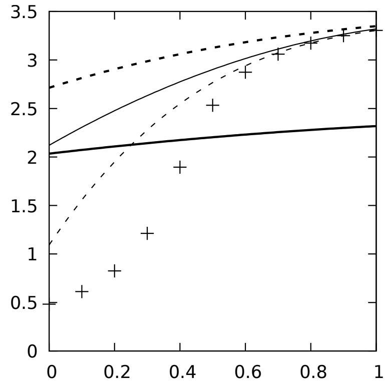
5.7 Numerical examples

In Figures 5.10 and 5.11 we show the performance of the truncated power series at the M-th order coefficient centered around $(1, 0)$. The $\sim$ is used to denote the polynomials that are calculated using the symmetric system (i.e. power series about $(\beta_1, \beta_2) = (1, 1)$) and are drawn in thicker lines. The setting is exactly the same as in Figures 5.3 and 5.4. The observations are also analogous. We have good performance near the expansion points see the top left and bottom right graph of Figure 5.10 and the bottom right graph of Figure 5.11.

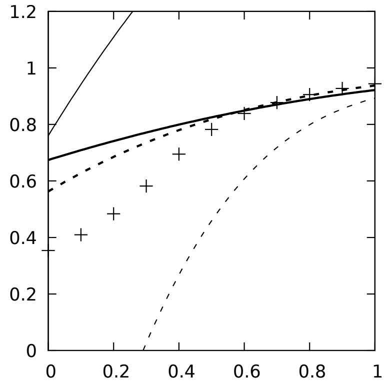
Now that we have power series approximations for every vertex of the (β_1, β_2) space, the task of combining them in one single approximation that is accurate for the complete space is posed. We can easily think of an extension of the Padé approximants in this two-dimensional setting, these are rational functions with two-dimensional polynomials in numerator and denominator. However,



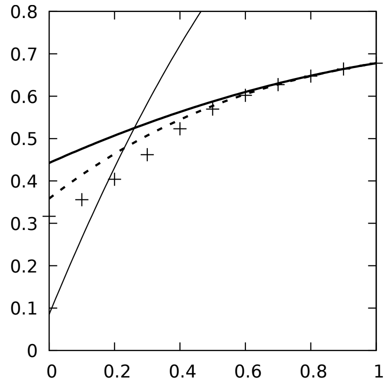
(a) $\bar{w}_2(\beta_1, 0)$



(b) $\bar{w}_2(\beta_1, 0.2)$

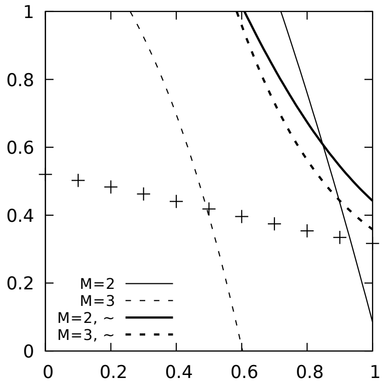


(c) $\bar{w}_2(\beta_1, 0.8)$

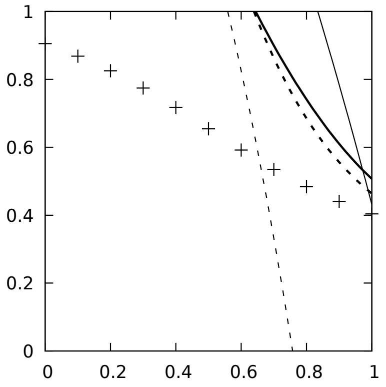


(d) $\bar{w}_2(\beta_1, 1)$

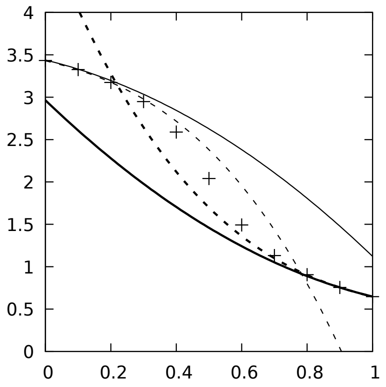
Figure 5.10: Truncated power series for $\bar{w}_2(\beta_1, \beta_2)$ with $N = 16$, $\lambda_T = 0.9, \alpha_1 = 0.4, \alpha_2 = 0.3$. The $\sim$ indicates the cases where we used the symmetric system.



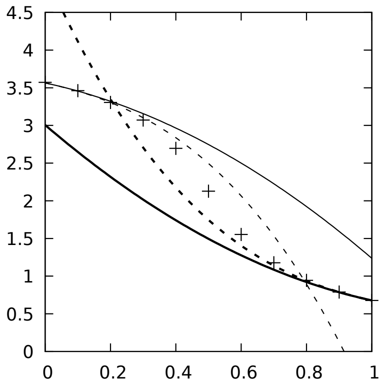
(a) $\bar{w}_2(0, \beta_2)$



(b) $\bar{w}_2(0.2, \beta_2)$



(c) $\bar{w}_2(0.8, \beta_2)$

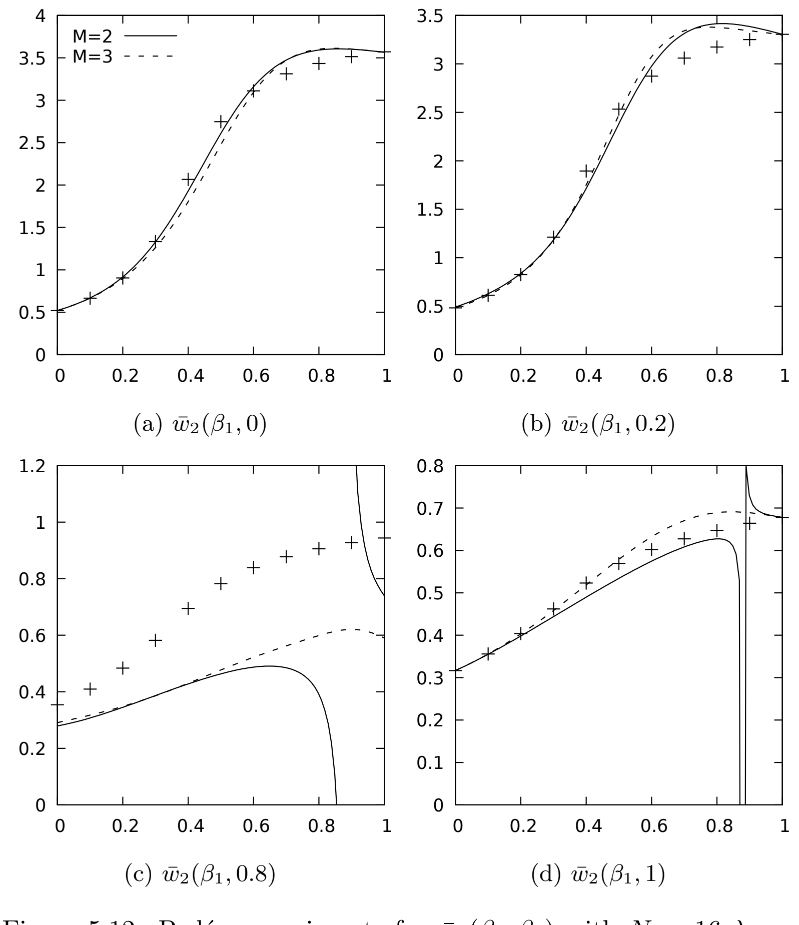


(d) $\bar{w}_2(1, \beta_2)$

Figure 5.11: Truncated power series for $\bar{w}_2(\beta_1, \beta_2)$ with $N = 16$, $\lambda_T = 0.9, \alpha_1 = 0.4, \alpha_2 = 0.3$. The $\sim$ indicates the cases where we used the symmetric system.

these two-dimensional Padé approximants are non-trivial to work with. A first big challenge is that much more unknowns are involved in determining the Padé approximants, which makes it computationally intensive and quickly intractable. For instance if, we use the four different power series truncated at $M = m + n = 2$, i.e., with constant term and terms of order $\beta_1, \beta_2, \beta_1^2, \beta_1\beta_2, \beta_2^2$, we thus have 24 parameters. As such, we require a Padé approximant with 24 unknowns. The second challenge is that the notion of singularities for two-dimensional functions is much more complex. Algorithm 4.1 is thus not simply extended for the two-dimensional functions, as the filtering of valid approximations is non-trivial.

The derivation of a good approximation for the complete (β_1, β_2) space deserves a study on its own. As such, we do not explore this line of research further in this dissertation. As a teaser, we did build a Padé approximant with 15 coefficients in the numerator and 9 in the denominator, that match the 24 coefficients of the four power series truncated at $M = 2$. Another Padé approximant with 21 coefficients in the numerator and 19 in the denominator was calculated from the truncated power series at $M = 3$. The result of these approximants is shown in Figures 5.12 and 5.13, together with the simulated results for comparison. For the Padé approximant with $M = 2$, we clearly see the consequences of the possibility of having singularities in the Padé approximant, i.e. some of the graphs show that the approximant has a pole. By coincidence the Padé approximant for $M = 3$ has no singularities, furthermore, we see that in most cases the approximant is reasonably accurate.



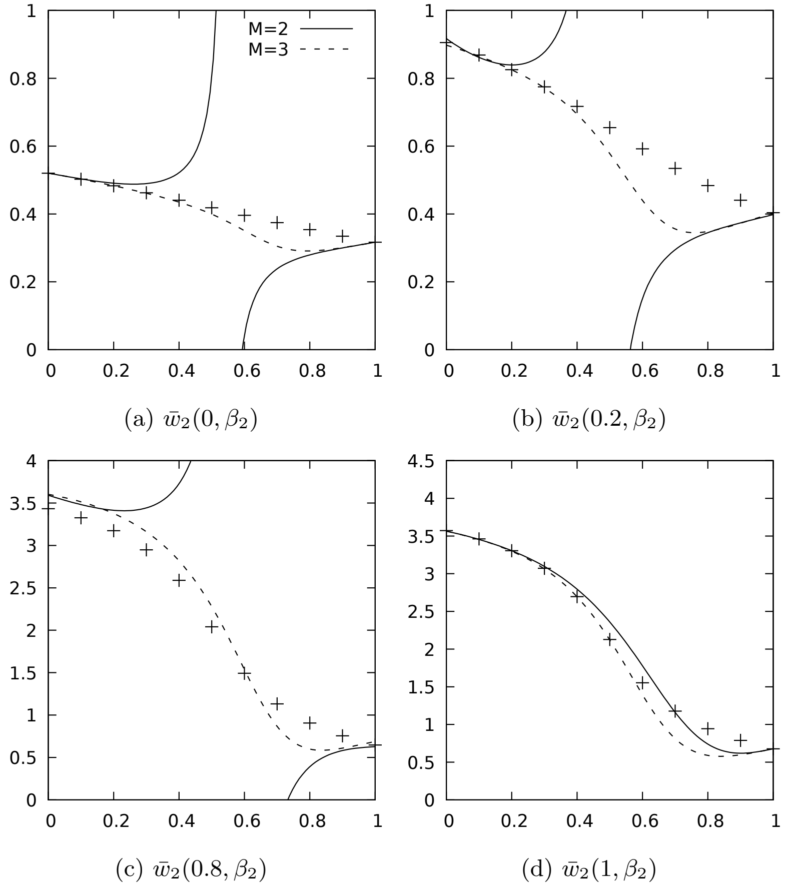
(a) $\bar{w}_2(\beta_1, 0)$

(b) $\bar{w}_2(\beta_1, 0.2)$

(c) $\bar{w}_2(\beta_1, 0.8)$

(d) $\bar{w}_2(\beta_1, 1)$

Figure 5.12: Padé approximants for $\bar{w}_2(\beta_1, \beta_2)$ with $N = 16, \lambda_T = 0.9, \alpha_1 = 0.4, \alpha_2 = 0.3$.



(a) $\bar{w}_2(0,\beta_2)$

(b) $\bar{w}_2(0.2,\beta_2)$

(c) $\bar{w}_2(0.8,\beta_2)$

(d) $\bar{w}_2(1,\beta_2)$

Figure 5.13: Padé approximants for $\bar{w}_2(\beta_1,\beta_2)$ with $N = 16, \lambda_T = 0.9, \alpha_1 = 0.4, \alpha_2 = 0.3$.

“Stel niet uit tot morgen wat je heden kunt doen.“

— Marijke Pattyn

6 Direct calculation of mean performance measures

In the previous two chapters, we have calculated a power series for the mean unfinished work of a GPS queueing system. This power series was derived from the power series of the joint pgf of the unfinished work in the queues. The conversion from a power series of the pgf to a power series of the means led to heavy use of l'Hôpital's rule. Together with the recursive nature of the power series this led to a limitation on the number of calculable coefficients in the power series. In this chapter, we therefore study the two-class model again but try to directly calculate the means from the system equations without passing through the probability generating functions.

In order to not overcomplicate the analysis, we restrict the arrival process to a bivariate Bernoulli process, that we summarize as follows:

$$\Pr[a_{1,k} = m, a_{2,k} = n] = p_{mn},$$
$$p_{00} + p_{01} + p_{10} + p_{11} = 1,$$

whereby $m, n = 1, 2$ and $a_{j,k}$ denotes the number of class-j ($j = 1, 2$) arrivals in slot k. $\Pr[a_{1,k} = m, a_{2,k} = n]$ with m or n greater than 1 equals zero. The total load is equal to $p_{01} + p_{10} + 2p_{11}$. Note that the restriction of the arrival process to this special case does not eliminate the major difficulties in the analysis of the GPS-model, but rather allows us to focus on finding solutions for these difficulties.

We start from the system equations and directly transform them to expressions for the mean values of the unfinished work. The expressions for the mean unfinished work are not fully specified, they contain unknown probabilities. By expanding all unknowns as their power series in β, we derive an iterative procedure to calculate the coefficients in the power series of the mean unfinished work. This is mainly done in Section 6.1. To eliminate all unknowns, the power series coefficients of some joint unfinished work probabilities are also required. These are calculated from the state diagram in Section 6.2. We also prove that the probability of having l customers in queue 2 is of $\mathrm{O}(\beta^{l-1})$ for $l > 0$. This proof is key to the solution and resulting algorithm.

In Section 6.3, we evaluate the numerical complexity of the resulting algorithm. Calculating the power series of the mean unfinished work of a certain class up to the M-th coefficient is of $\mathrm{O}(M^3)$. This method certainly allows us to calculate more coefficients in the series than our previous method via the generating function. Having more coefficients enables us to estimate the dominant singularities limiting the convergence radius of the series. In return, combining the knowledge of these singularities and having more coefficients allows us to build better approximations. In general, the area of convergence may not include the complete interval $[0, 1]$. Therefore, the utility of the approximation has to be extended outside the convergence area of the power series to cover the whole interval of interest $\beta = [0, 1]$. Building this approximation and evaluating its performance is done in Section 6.4.

6.1 Analysis mean unfinished work

In this section, we derive equations for the mean unfinished work in both queues starting from the system equations. These equations entail partials of these means and as a result are not directly solvable. By writing all unknowns as their power series in β, we show that it is possible to solve for the coefficients of interest. The end result of this section is an iterative procedure to calculate higher-order coefficients of the mean unfinished work of a certain class. The expressions for these coefficients are a function of coefficients of unknown probabilities; eliminating these is the subject of the next section.

We write the mean unfinished work of queue j as: $\mathrm{E}[w_j] = \sum_{n=0}^{\infty} \mathrm{E}[w_j]_n \beta^n$. With $\mathbb{1}_C$ defined as the indicator function which equals 1 if condition C is true and 0 otherwise, we formulate our first theorem.

Theorem 6.1. *The coefficients for the power series of the mean unfinished work satisfy:*

$$-(p_{10}+p_{11})\mathrm{E}[w_1]_n = \begin{cases} (p_{10}+p_{11})(1-p_{10}-p_{11}) - \mathrm{E}[w_1\mathbb{1}_{w_2=0}]_0, & n=0 \\ -\mathrm{E}[w_1]_{n-1} - \mathrm{E}[w_1\mathbb{1}_{w_2=0}]_n + \mathrm{E}[w_1\mathbb{1}_{w_2=0}]_{n-1}, & \textit{otherwise} \end{cases}$$

$$(1-p_{01}-p_{11})\mathrm{E}[w_2]_n = \begin{cases} (p_{01}+p_{11})(1-p_{01}+p_{11}), & n=0 \\ \mathrm{E}[w_2]_{n-1} - \mathrm{E}[w_2\mathbb{1}_{w_1=0}]_{n-1}, & \textit{otherwise} \end{cases}$$

Proof. We start by reformulating the system equations for the unfinished work that were listed in Section 1.3, using the indicator function. We have

$$\begin{cases} w_{1,k+1} &= a_{1,k} + w_{1,k} - \mathbb{1}_{w_{1,k}>0, w_{2,k}=0} - \mathbb{1}_{w_{1,k}>0, w_{2,k}>0, r_k<\beta} \\ w_{2,k+1} &= a_{2,k} + w_{2,k} - \mathbb{1}_{w_{1,k}=0, w_{2,k}>0} - \mathbb{1}_{w_{1,k}>0, w_{2,k}>0, r_k>\beta}. \end{cases} \tag{6.1}$$

In these equations r_k denotes the decision variable that we used before and that is generated in slot k from a uniform distribution in $[0,1]$. When $r_k \leq \beta$ queue 1 is served (i.e. with probability β) and when $r_k > \beta$ queue 2 is served (i.e. with probability $1-\beta$), given that both classes are backlogged ($w_{1,k} > 0, w_{2,k} > 0$).

Taking the limit of (6.1) for $k \to \infty$ and calculating the expected value of both sides, eliminates $\mathrm{E}[w_j]$. This step does therefore not lead to expressions for the mean unfinished work in both queues in steady state directly, they do however provide some valuable equations:

$$\begin{aligned} 0 &= (p_{10}+p_{11}) - \Pr[w_1>0, w_2=0] - \beta\Pr[w_1>0, w_2>0], \quad (6.2) \\ 0 &= (p_{01}+p_{11}) - \Pr[w_1=0, w_2>0] - (1-\beta)\Pr[w_1>0, w_2>0]. \end{aligned}$$

Secondly, we take the expected values of the squares of both sides of equations (6.1). The first equation yields:

$$\begin{aligned} \mathrm{E}[w_1^2] =& \mathrm{E}[(a_1 + w_1 - \mathbb{1}_{w_1>0,w_2=0} - \mathbb{1}_{w_1>0,w_2>0,r<\beta})^2] \\ =& \mathrm{E}[w_1^2] + \mathrm{E}[a_1^2] + \Pr[w_1>0, w_2=0] + \beta\Pr[w_1>0, w_2>0] \\ &+ 2\mathrm{E}[a_1]\mathrm{E}[w_1] - 2\mathrm{E}[w_1\mathbb{1}_{w_2=0}] - 2\beta\mathrm{E}[w_1\mathbb{1}_{w_2>0}] \\ &- 2\mathrm{E}[a_1]\Pr[w_1>0, w_2=0] - 2\mathrm{E}[a_1]\beta\Pr[w_1>0, w_2>0]. \end{aligned}$$

Eliminating $\mathrm{E}[w_1^2]$ and using (6.2), we find

$$\begin{aligned} &(\beta - p_{10} - p_{11})\mathrm{E}[w_1] \quad (6.3) \\ &\qquad = (p_{10}+p_{11})(1-p_{10}-p_{11}) - (1-\beta)\mathrm{E}[w_1\mathbb{1}_{w_2=0}]. \end{aligned}$$

Analogous for queue 2, we get from the second equation of (6.1):

$$\begin{aligned}(1-\beta-p_{01}-&p_{11})\mathrm{E}[w_2] \\ &=(p_{01}+p_{11})(1-p_{01}+p_{11})-\beta\mathrm{E}[w_2\mathbb{1}_{w_1=0}].\end{aligned} \tag{6.4}$$

Lastly, we write $\mathrm{E}[w_j]$ and $\mathrm{E}[w_j\mathbb{1}_{w_{3-j}=0}]$ as a power series in β, i.e.

$$\mathrm{E}[w_j]=\sum_{n=0}^{\infty}\mathrm{E}[w_j]_n\beta^n,$$
$$\mathrm{E}[w_j\mathbb{1}_{w_{3-j}=0}]=\sum_{n=0}^{\infty}\mathrm{E}[w_j\mathbb{1}_{w_{3-j}=0}]_n\beta^n.$$

Doing so, and by equating equal order coefficients in β in (6.3)-(6.4), we find the equations from the theorem. □

In the remainder, we further concentrate on $\mathrm{E}[w_2]$. As was discussed in the previous chapters extensively, $\mathrm{E}[w_1]$ can be calculated from the total unfinished work $\mathrm{E}[w_T]$, or by considering the symmetry of the system. Yet another alternative, of course, is to extend the remainder of this chapter to calculate $\mathrm{E}[w_1]$ which is a straightforward exercise.

Theorem 6.1 provides an equation for $\mathrm{E}[w_2]_n$ but contains the unknown partial mean $\mathrm{E}[w_2\mathbb{1}_{w_1=0}]_n$. Given (the power series of) $\mathrm{E}[w_2\mathbb{1}_{w_1=0}]$, the power series of $\mathrm{E}[w_2]$ can be calculated iteratively using Theorem 6.1. Calculating the coefficients of this partial mean is the subject of our next theorem.

Theorem 6.2. *The coefficients of the partial mean* $\mathrm{E}[w_2\mathbb{1}_{w_1=i}]$ *satisfy the expressions on page 173.*

Proof. To prove (6.6), we first list the system equation for $w_{2,k+1}$ given that $w_{1,k+1}=0$:

$$w_{2,k+1} = \begin{cases} w_{2,k}+a_{2,k}-\mathbb{1}_{w_{2,k}>0}; & w_{1,k}=0, a_{1,k}=0 \\ a_{2,k}; & w_{1,k}=1, a_{1,k}=0, w_{2,k}=0 \\ w_{2,k}+a_{2,k}; & w_{1,k}=1, a_{1,k}=0, w_{2,k}>0, r_k<\beta \end{cases}$$

If in slot $k+1$ queue 1 is empty, then in slot k queue 1 was either empty and had no arrivals (case 1) or had one customer that got served in that slot and no arrivals (cases 2 and 3). If queue 1 contained one customer in slot k it gets served either because queue 2 was empty (case 2) or because of the probabilistic decision of the server (case 3).

Theorem 6.2. *The coefficients of the partial mean* $\mathrm{E}[w_2 \mathbb{1}_{w_1=i}]$ *satisfy:*

- $i = 0$:

$$\begin{aligned}(1-p_{00}-p_{01})\mathrm{E}[w_2\mathbb{1}_{w_1=0}]_0 &= -\,p_{00}\Pr[w_1=0]_0 + (p_{00}+p_{01})\Pr[w_1=0,w_2=0] + p_{01}\Pr[w_1=1,w_2=0]_0,\\ (1-p_{00}-p_{01})\mathrm{E}[w_2\mathbb{1}_{w_1=0}]_n &= -\,p_{00}\Pr[w_1=0]_n + (p_{00}+p_{01})\mathrm{E}[w_2\mathbb{1}_{w_1=1}]_{n-1}\\ &\qquad + p_{01}\big(\Pr[w_1=1,w_2=0]_n - \Pr[w_1=1,w_2=0]_{n-1} + \Pr[w_1=1]_{n-1}\big),\end{aligned} \tag{6.6}$$

- $i = 1$:

$$\begin{aligned}(1-p_{00}-p_{01})\mathrm{E}[w_2\mathbb{1}_{w_1=1}]_0 &= (p_{00}+p_{11})\Pr[w_1=1,w_2=0]_0 + p_{01}\Pr[w_1=2,w_2=0]_0\\ &\quad + (p_{10}+p_{11})\big(\mathrm{E}[w_2\mathbb{1}_{w_1=0}]_0 + \Pr[w_1=0,w_2=0]\big) + p_{10}\Pr[w_1=0]_0 - p_{00}\Pr[w_1=1]_0,\\ (1-p_{00}-p_{01})\mathrm{E}[w_2\mathbb{1}_{w_1=1}]_n &= (p_{00}+p_{11})\big(\Pr[w_1=1,w_2=0]_n - \Pr[w_1=1,w_2=0]_{n-1} + \Pr[w_1=1]_{n-1}\big)\\ &\quad + (p_{10}+p_{11})\big(\mathrm{E}[w_2\mathbb{1}_{w_1=0}]_n + \mathrm{E}[w_2\mathbb{1}_{w_1=1}]_{n-1}\big) + (p_{01}+p_{00})\big(\mathrm{E}[w_2\mathbb{1}_{w_1=2}]_{n-1} - \mathrm{E}[w_2\mathbb{1}_{w_1=1}]_{n-1}\big)\\ &\quad - p_{00}\Pr[w_1=1]_n + p_{10}\Pr[w_1=0]_n + p_{01}\big(\Pr[w_1=2]_{n-1} + \Pr[w_1=2,w_2=0]_n - \Pr[w_1=2,w_2=0]_{n-1}\big),\end{aligned} \tag{6.7}$$

- $i > 1$:

$$\begin{aligned}(1-p_{00}-p_{01})\mathrm{E}[w_2\mathbb{1}_{w_1=i}]_0 &= p_{10}\big(\Pr[w_1=i-1,w_2=0]_0 - \Pr[w_1=i-1]_0\big)\\ &\quad + (p_{00}+p_{11})\Pr[w_1=i,w_2=0]_0 + p_{01}\Pr[w_1=i+1,w_2=0]_0 + (p_{10}+p_{11})\mathrm{E}[w_2\mathbb{1}_{w_1=i-1}]_0 - p_{00}\Pr[w_1=i]_0,\\ (1-p_{00}-p_{01})\mathrm{E}[w_2\mathbb{1}_{w_1=i}]_n &= p_{01}\big(\Pr[w_1=i+1]_{n-1} + \Pr[w_1=i+1,w_2=0]_n - \Pr[w_1=i+1,w_2=0]_{n-1}\big)\\ &\quad + (p_{10}+p_{11})\big(\mathrm{E}[w_2\mathbb{1}_{w_1=i-1}]_n - \mathrm{E}[w_2\mathbb{1}_{w_1=i-1}]_{n-1} + \mathrm{E}[w_2\mathbb{1}_{w_1=i}]_{n-1}\big) + (p_{01}+p_{00})\big(\mathrm{E}[w_2\mathbb{1}_{w_1=i+1}]_{n-1} - \mathrm{E}[w_2\mathbb{1}_{w_1=i}]_{n-1}\big)\\ &\quad + (p_{00}+p_{11})\big(\Pr[w_1=i,w_2=0]_n - \Pr[w_1=i,w_2=0]_{n-1} + \Pr[w_1=i]_{n-1}\big) - p_{00}\Pr[w_1=i]_n\\ &\quad + p_{10}\big(\Pr[w_1=i-1,w_2=0]_n - \Pr[w_1=i-1,w_2=0]_{n-1} + \Pr[w_1=i-1]_{n-1} - \Pr[w_1=i-1]_n\big),\end{aligned} \tag{6.8}$$

with $n > 0$.

By using the same steps as in the proof of Theorem 6.1 we find:

$$\begin{aligned}(1 - p_{00} - p_{01})\mathrm{E}[w_2 \mathbb{1}_{w_1=0}] \\ &= - p_{00} \Pr[w_1 = 0] \\ &\quad + (p_{00} + p_{01})\big(\Pr[w_1 = 0, w_2 = 0] + \beta \mathrm{E}[w_2 \mathbb{1}_{w_1=1}]\big) \\ &\quad + p_{01}\big(\beta \Pr[w_1 = 1] + (1 - \beta) \Pr[w_1 = 1, w_2 = 0]\big).\end{aligned} \tag{6.5}$$

The probability of a completely empty system, i.e. $\Pr[w_1 = 0, w_2 = 0]$, is independent of β and equals $1 - \lambda_T = 1 - p_{01} - p_{10} - 2p_{11}$. After expanding all performance measures in (6.5) as power series in β and equating equal order coefficients (6.6) is obtained.

The parts of the proof for (6.7) and (6.8) are analogous to the part for (6.6), and are therefore omitted. □

Theorem 6.2 provides equations for the power series of $\mathrm{E}[w_2 \mathbb{1}_{w_1=0}]$ in (6.6). This however introduces the unknown $\mathrm{E}[w_2 \mathbb{1}_{w_1=1}]$. As such (6.7) is included which subsequently introduces $\mathrm{E}[w_2 \mathbb{1}_{w_1=2}]$ and so on. Note that (6.7) cannot be included in (6.8). Setting $i = 1$ in (6.8) produces different (and wrong) equations than in (6.7). This is because the empty state, when both queues are empty comes in to play in (6.7), i.e. considering state transitions to and from states with 1 customer in queue 1. From the equations in the theorem, we see the order of the needed coefficients reduces, but higher probabilities are involved.

Using the theorems presented in this section iteratively, all expectations ($\mathrm{E}[\cdot]$) can be written as partial expectations of the known priority case ($\beta = 0$), i.e. the 0-th order coefficient. However, the expressions still contain coefficients of probabilities (for instance of: $\Pr[w_1 = 0]$, $\Pr[w_1 = 1]$, $\Pr[w_1 = 1, w_2 = 0]$, $\ldots$) that we have not yet calculated. The calculation of these coefficients is the subject of the next section.

6.2 Analysis of stationary unfinished work probabilities

In the first part of this section, we calculate probabilities of the form $\Pr[w_1 = \cdot]$. The process and result is similar to the procedure we used to calculate $\mathrm{E}[w_2 \mathbb{1}_{w_1=\cdot}]$ in the previous section. The resulting expressions contain joint probabilities of the unfinished work, the solution of which is the subject of the second part of this section. To this end, we construct and analyze the state diagram of the queueing system. From the state diagram, we prove that numerous power series coefficients of the state probabilities are zero. Using this observation the last remaining unknowns are eliminated from the expressions for the coefficients of $\mathrm{E}[w_2]$, producing a well-defined iterative algorithm for their calculation.

We start with $\Pr[w_1 = i]$:

Theorem 6.3. *The coefficients of the probability* $\Pr[w_1 = i]$ *satisfy the expressions on page 177.*

Proof. The proof is analogous to the proof of Theorem 6.2. For (6.9), we start from the system equation (6.1) and do a probability transformation.

$$\begin{aligned}
&\Pr[w_{1,k+1} = 0]\\
&\qquad = \Pr[a_{1,k} + w_{1,k} - \mathbb{1}_{w_{1,k}>0, w_{2,k}=0} - \mathbb{1}_{w_{1,k}>0, w_{2,k}>0, r_k<\beta} = 0]\\
&\qquad = (p_{00} + p_{01})\big(\Pr[w_{1,k} = 0] + \Pr[w_{1,k} = 1, w_{2,k} = 0]\\
&\qquad\qquad + \beta \Pr[w_{1,k} = 1, w_{2,k} > 0]\big)
\end{aligned}$$

Taking the limit for $k \to \infty$ (steady state), isolating $\Pr[w_1 = 0]$ and using the equality $\Pr[w_1 = 1, w_2 > 0] = \Pr[w_1 = 1] - \Pr[w_1 = 1, w_2 = 0]$, we find:

$$\begin{aligned}
&(1 - p_{00} - p_{01}) \Pr[w_1 = 0]\\
&\qquad = (p_{00} + p_{01})\big(\Pr[w_1 = 1, w_2 = 0]\\
&\qquad\qquad + \beta(\Pr[w_1 = 1] - \Pr[w_1 = 1, w_2 = 0])\big)
\end{aligned}$$

Lastly, we expand the probabilities as power series in β and compare terms of equal powers of β. This produces the expressions from (6.9). Again, to prove equations (6.10) and (6.11) the reasoning is analogous. □

The remaining unknowns in the previous theorems are the coefficients of joint probabilities $\Pr[w_1 = i, w_2 = 0]$ ($i \geq 1$). In the remainder of this section, we show how to calculate these. We first construct the state diagram of the discrete-time Markov chain $\{(w_{1,k}, w_{2,k})\}$ in Figure 6.1. The state diagram continues infinitely to the right and bottom of the figure. State $(0, 1)$ signifies the state where $w_1 = 0$ and $w_2 = 1$. To avoid overloading the image, we have omitted the reflexive transitions; for instance state $(0, 1)$ should include a loop to itself with transition probability p_{01} so as to make all outgoing transitions sum to 1. Solving the balance equations for $\Pr[w_1 = i, w_2 = 0]$ is equivalent to solving for all state probabilities of the state diagram. Consequently, we consider the whole state diagram in our

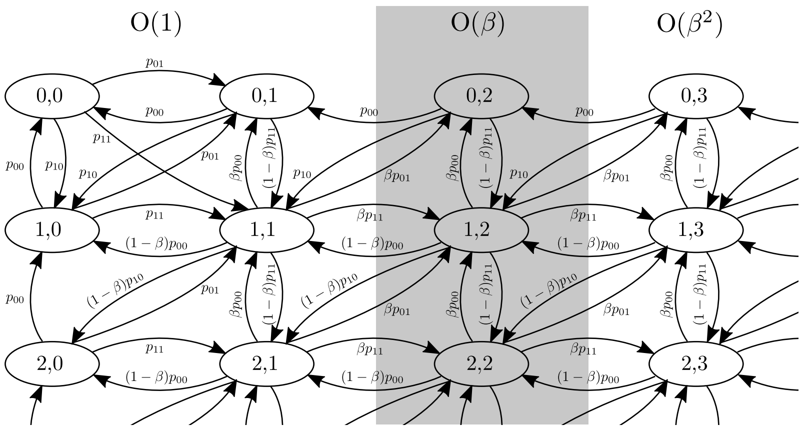


Figure 6.1: State diagram, omitting reflexive transitions

Theorem 6.3. *The coefficients of the probability* $\Pr[w_1 = i]$ *satisfy:*

- $i = 0$*:*

$$\begin{aligned}(1 - p_{00} - p_{01}) \Pr[w_1 = 0]_0 &= (p_{00} + p_{01}) \Pr[w_1 = 1, w_2 = 0]_0, \\ (1 - p_{00} - p_{01}) \Pr[w_1 = 0]_n &= (p_{00} + p_{01}) \big(\Pr[w_1 = 1, w_2 = 0]_n - \Pr[w_1 = 1, w_2 = 0]_{n-1} + \Pr[w_1 = 1]_{n-1} \big),\end{aligned} \tag{6.9}$$

- $i = 1$*:*

$$\begin{aligned}&(1 - p_{00} - p_{01}) \Pr[w_1 = 1]_0 \\ &\quad = (p_{00} + p_{01}) \Pr[w_1 = 2, w_2 = 0]_0 + (1 - 2(p_{00} + p_{01})) \Pr[w_1 = 1, w_2 = 0]_0 + (1 - p_{00} - p_{01}) \Pr[w_1 = 0]_0, \\ &(1 - p_{00} - p_{01}) \Pr[w_1 = 1]_n \\ &\quad = (p_{00} + p_{01}) \big(\Pr[w_1 = 2, w_2 = 0]_n - \Pr[w_1 = 2, w_2 = 0]_{n-1} + \Pr[w_1 = 2]_{n-1} \big) + (1 - p_{00} - p_{01}) \Pr[w_1 = 0]_n \\ &\qquad + (1 - 2(p_{00} + p_{01})) \big(\Pr[w_1 = 1]_{n-1} + \Pr[w_1 = 1, w_2 = 0]_n - \Pr[w_1 = 1, w_2 = 0]_{n-1} \big),\end{aligned} \tag{6.10}$$

- $i > 1$*:*

$$\begin{aligned}&(1 - p_{00} - p_{01}) \Pr[w_1 = i]_0 \\ &\quad = (p_{00} + p_{01}) \Pr[w_1 = i + 1, w_2 = 0]_0 + (1 - 2(p_{00} + p_{01})) \Pr[w_1 = i, w_2 = 0]_0 \\ &\qquad + (1 - p_{00} - p_{01}) \big(\Pr[w_1 = i - 1]_0 - \Pr[w_1 = i - 1, w_2 = 0]_0 \big), \\ &(1 - p_{00} - p_{01}) \Pr[w_1 = i]_n \\ &\quad = (p_{00} + p_{01}) \big(\Pr[w_1 = i + 1, w_2 = 0]_n - \Pr[w_1 = i + 1, w_2 = 0]_{n-1} + \Pr[w_1 = i + 1]_{n-1} \big) \\ &\qquad + (1 - 2(p_{00} + p_{01})) \big(\Pr[w_1 = i]_{n-1} + \Pr[w_1 = i, w_2 = 0]_n - \Pr[w_1 = i, w_2 = 0]_{n-1} \big) \\ &\qquad + (1 - p_{00} - p_{01}) \big(\Pr[w_1 = i - 1]_n - \Pr[w_1 = i - 1]_{n-1} + \Pr[w_1 = i - 1, w_2 = 0]_n - \Pr[w_1 = i - 1, w_2 = 0]_{n-1} \big).\end{aligned} \tag{6.11}$$

with $n > 0$.

search for $\Pr[w_1 = i, w_2 = 0]$. We see from the figure that all transitions from states $(\cdot, l-1)$ to $(\cdot, l)$ with $l > 1$ possess a factor β in their transition probability. As a result the steady state probability of $\Pr[w_1 = \cdot, w_2 = l]$ is of $\mathrm{O}(\beta^{l-1})$.

Lemma 6.1. $\Pr[w_1 = \cdot, w_2 = l]$ *is of* $O(\beta^{l-1})$ *with* $l \geq 1$ *and* $\Pr[w_1 = \cdot, w_2 = 0] = O(1)$.

The lemma is intuitively understood looking at sample paths to arrive in state $w_1 = \cdot, w_2 = l$. When $\beta = 0$ class 2 has strict priority and $w_2 \leq 1$, as at most one packet of each class can arrive in the same slot (Bernoulli arrivals). For queue 2 to contain l packets the scheduler had to deviate from the strict priority scheduling minimally $l-1$ times, each time with probability β, which accounts for the order of $\Pr[w_1 = \cdot, w_2 = l]$. This intuitive reasoning is known in the literature as the n-events rule, stating that for the n-th order coefficient only sample paths with n or fewer of such events must be considered [25, 47, 117]. The *event* in our context refers to the server making a decision and choosing queue 1 (which happens with probability β). We now formalize the proof.

Proof. This lemma states that $\Pr[w_1 = \cdot, w_2 = l]$ is of $\mathrm{O}(\beta^{l-1})$ for $l > 1$. In concreto, this means $\Pr[w_1 = \cdot, w_2 = l]_n = 0$ for $l > 1, n < l-1$. To this end, we construct the balance equations regarding column transitions in the state diagram shown in Figure 6.1. For the transition from $l-1$-th to l-th column ($l > 1$), we get:

$$\begin{aligned}(p_{01} + p_{11})\beta\big(&\Pr[w_2 = l-1] - \Pr[w_1 = 0, w_2 = l-1]\big) \qquad (6.12)\\ =&(p_{00} + p_{10})\Big(\Pr[w_1 = 0, w_2 = l]\\ &+ (1-\beta)\big(\Pr[w_2 = l] - \Pr[w_1 = 0, w_2 = l]\big)\Big).\end{aligned}$$

Subsequently, we write all probabilities as power series in β and equate equal order coefficients. Now we study the resulting equations inductively in l, proving the lemma. For the base case, we consider the constant coefficient of (6.12) with $l = 2$:

$$0 = (p_{00} + p_{10}) \Pr[w_2 = 2]_0.$$

As $\Pr[w_2 = 2]_0 = \sum_{i=0}^{\infty} \Pr[w_1 = i, w_2 = 2]_0$ and the constant coefficient of $\Pr[w_1 = i, w_2 = 2]$ is non-negative (being the probability in the case $\beta = 0$), $\Pr[w_1 = i, w_2 = 2]_0 = 0$. Therefore $\Pr[w_1 = i, w_2 = 0] = \mathrm{O}(\beta)$ and this ends the proof of base case.

The induction hypothesis states $\Pr[w_1 = \cdot, w_2 = l']_n = 0$ for $l' > 1, n < l' - 1$ and $l' < l$. In this inductive step, we consider the coefficients in (6.12). For the constant coefficient, this gives:

$$0 = (p_{00} + p_{10}) \Pr[w_2 = l]_0,$$

proving $\Pr[w_1 = \cdot, w_2 = l]_0 = 0$. For the coefficient of $\beta^n, 0 < n < l - 1$, we write:

$$\begin{aligned}(p_{01} + p_{11})\Big(\Pr[w_2 = l-1]_{n-1} - \Pr[w_1 = 0, w_2 = l-1]_{n-1}\Big)\\ =&(p_{00} + p_{10})\Big(\Pr[w_2 = l]_n - \Pr[w_2 = l]_{n-1}\\ &+ \Pr[w_1 = 0, w_2 = l]_{n-1}\Big).\end{aligned}$$

The left hand side equals zero as a result of the induction hypothesis. For $n = 1$, the last two terms of the right-hand side ($\Pr[w_2 = l]_0$, $\Pr[w_1 = 0, w_2 = l]_0$) are already known to be 0. Consequently, $\Pr[w_2 = l]_1 = 0 = \sum_{i=0}^{\infty} \Pr[w_1 = i, w_2 = l]_1$. Furthermore, $\Pr[w_1 = \cdot,$ $w_2 = l]_1$ should be positive or zero, as smaller order coefficients equal zero and the first non-zero coefficient determines the sign of the power series for infinitesimal β, which should be positive being a probability. As such, $\Pr[w_1 = \cdot, w_2 = l]_1 = 0$. This reasoning can iteratively be continued for $n < l - 1$, proving $\Pr[w_1 = \cdot, w_2 = l]_n = 0$ for $l > 1, n < l - 1$. □

As a consequence of this lemma numerous coefficients of the power series in β of the state probabilities equal zero. In Figure 6.2, we show a 3-dimensional plot indicating with crosses where $\Pr[w_1 = i, w_2 = l]_n$ is non-zero.

We now consider the balance equations for each state in the state diagram. We observe that for the balance equation of each state in row i of the state diagram all terms including $\Pr[w_1 = i + 1, w_2 = \cdot]$ have a factor β except for $\Pr[w_1 = i + 1, w_2 = 0]$. After expanding all probabilities as power series in β and equating equal order coefficients, we end up with equations for $\Pr[w_1 = i, w_2 = l]_n$ ($l > 0$) and $\Pr[w_1 = i + 1, w_2 = 0]_n$ only containing coefficients $\Pr[w_1 = i', w_2 = l]_{n'}$ of either lower order ($n' < n$) or on a higher row in the state diagram ($i' \leq i$). As the amount of non-zero coefficients for β^n of the probabilities of states on the same row is finite, we can iteratively calculate equal-order coefficients row-by-row starting from the top using coefficients from a lower order. This effectively sums up an iterative procedure to calculate all power series coefficients for the state probabilities of the state diagram. We list the resulting expressions on pages 181-182.

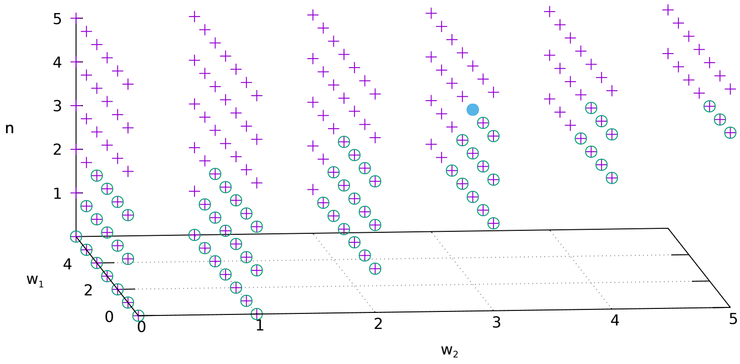


Figure 6.2: Diagram of non-zero coefficients $\Pr[w_1 = i, w_2 = l]_n$ (crosses), the circled coefficients are needed to calculate $\Pr[w_1 = 2, w_2 = 3]_4$ indicated by the solid disk.

$\underline{\Pr[w_1 = i+1, w_2 = 0] \text{ and } \Pr[w_1 = i, w_2 = 1]}$

- $i = 0$:

$$\Pr[w_1 = 0, w_2 = 1]_n = \frac{p_{01}}{p_{00}} \Pr[w_1 = 0, w_2 = 0]_n + p_{00}\big(\Pr[w_1 = 0, w_2 = 2]_n + \Pr[w_1 = 1, w_2 = 1]_{n-1}\big) \tag{6.13}$$

$$\Pr[w_1 = 1, w_2 = 0]_n = \frac{1 - p_{00} - p_{01}}{p_{00}} \Pr[w_1 = 0, w_2 = 0]_n - p_{00}\big(\Pr[w_1 = 0, w_2 = 2]_n + \Pr[w_1 = 1, w_2 = 1]_{n-1}\big)$$

- $i = 1$:

$$\begin{aligned}\Pr[w_1 = 2, w_2 = 0]_n = {} & \frac{-p_{10}(1-p_{01}) - p_{11}p_{00}}{p_{00}^2} \Pr[w_1 = 0, w_2 = 0]_n + (1-p_{10}) \Pr[w_1 = 1, w_2 = 1]_{n-1} \\ & + \frac{1-p_{01}}{p_{00}} \Pr[w_1 = 1, w_2 = 0]_n - p_{10} \Pr[w_1 = 0, w_2 = 2]_n \\ & - p_{00}\big(\Pr[w_1 = 1, w_2 = 2]_n - \Pr[w_1 = 1, w_2 = 2]_{n-1} + \Pr[w_1 = 2, w_2 = 1]_{n-1}\big)\end{aligned} \tag{6.14}$$

$$\begin{aligned}\Pr[w_1 = 1, w_2 = 1]_n = {} & \frac{p_{11}p_{00} - p_{10}p_{01}}{p_{00}^2} \Pr[w_1 = 0, w_2 = 0]_n + p_{10} \Pr[w_1 = 1, w_2 = 1]_{n-1} + p_{10} \Pr[w_1 = 0, w_2 = 2]_n \\ & + \frac{p_{01}}{p_{00}} \Pr[w_1 = 1, w_2 = 0]_n + p_{00}\big(\Pr[w_1 = 1, w_2 = 2]_n - \Pr[w_1 = 1, w_2 = 2]_{n-1} + \Pr[w_1 = 2, w_2 = 1]_{n-1}\big)\end{aligned}$$

- $i > 1$:

$$\begin{aligned}\Pr[w_1 = i, w_2 = 1]_n = {} & \frac{p_{11}p_{00} + p_{01}(1-p_{10})}{p_{00}} \Pr[w_1 = i, w_2 = 0]_n \\ & + \frac{p_{11}p_{00} - p_{10}p_{01}}{p_{00}} \big(\Pr[w_1 = i-1, w_2 = 1]_n - \Pr[w_1 = i-1, w_2 = 1]_{n-1}\big) \\ & + p_{00}\big(\Pr[w_1 = i, w_2 = 2]_n - \Pr[w_1 = i, w_2 = 2]_{n-1} + \Pr[w_1 = i+1, w_2 = 1]_{n-1}\big) \\ & + p_{10}\big(\Pr[w_1 = i-1, w_2 = 2]_n - \Pr[w_1 = i-1, w_2 = 2]_{n-1} + \Pr[w_1 = i, w_2 = 1]_{n-1}\big)\end{aligned} \tag{6.15}$$

$$\Pr[w_1 = i+1, w_2 = 0]_n = \frac{(1-p_{10})(1-p_{01}) - p_{11}p_{00}}{p_{00}} \Pr[w_1 = i, w_2 = 0]_n + (1-p_{10}) \Pr[w_1 = i, w_2 = 1]_{n-1}$$

$$+\frac{-p_{10}(1-p_{01})-p_{11}p_{00}}{p_{00}}\big(\Pr[w_1=i-1,w_2=1]_n-\Pr[w_1=i-1,w_2=1]_{n-1}\big)$$
$$-p_{00}\big(\Pr[w_1=i,w_2=2]_n-\Pr[w_1=i,w_2=2]_{n-1}+\Pr[w_1=i+1,w_2=1]_{n-1}\big)$$
$$-p_{10}\big(\Pr[w_1=i-1,w_2=2]_n-\Pr[w_1=i-1,w_2=2]_{n-1}\big)$$

<u>$\Pr[w_1=i,w_2=l]_n$ with $l>1$</u>

- $i=0$:

$$\Pr[w_1=0,w_2=l]_n=\frac{p_{01}}{1-p_{01}}\Pr[w_1=1,w_2=l-1]_{n-1}+\frac{p_{00}}{1-p_{01}}\big(\Pr[w_1=0,w_2=l+1]_n+\Pr[w_1=1,w_2=l]_{n-1}\big) \tag{6.16}$$

- $i=1$:

$$\begin{aligned}\Pr[w_1=1,w_2=l]_n=&\frac{p_{01}}{1-p_{01}}\Pr[w_1=2,w_2=l-1]_{n-1}+\frac{p_{10}-p_{01}}{1-p_{01}}\Pr[w_1=1,w_2=l]_{n-1}\\&+\frac{p_{00}}{1-p_{01}}\big(\Pr[w_1=2,w_2=l]_{n-1}+\Pr[w_1=1,w_2=l+1]_n-\Pr[w_1=1,w_2=l+1]_{n-1}\big)\\&+\frac{p_{10}}{1-p_{01}}\Pr[w_1=0,w_2=l+1]_n+\frac{p_{11}}{1-p_{01}}\big(\Pr[w_1=1,w_2=l-1]_{n-1}+\Pr[w_1=0,w_2=l]_n\big)\end{aligned} \tag{6.17}$$

- $i>1$:

$$\begin{aligned}\Pr[w_1=i,w_2=l]_n=&\frac{p_{01}}{1-p_{01}}\Pr[w_1=i+1,w_2=l-1]_{n-1}+\frac{p_{10}-p_{01}}{1-p_{01}}\Pr[w_1=i,w_2=l]_{n-1}\\&+\frac{p_{00}}{1-p_{01}}\big(\Pr[w_1=i+1,w_2=l]_{n-1}+\Pr[w_1=i,w_2=l+1]_n-\Pr[w_1=i,w_2=l+1]_{n-1}\big)\\&+\frac{p_{10}}{1-p_{01}}\big(\Pr[w_1=i-1,w_2=l+1]_n-\Pr[w_1=i-1,w_2=l+1]_{n-1}\big)\\&+\frac{p_{11}}{1-p_{01}}\big(\Pr[w_1=i,w_2=l-1]_{n-1}+\Pr[w_1=i-1,w_2=l]_n-\Pr[w_1=i-1,w_2=l]_{n-1}\big)\end{aligned} \tag{6.18}$$

To illustrate the construction of these expressions, we provide a small example of the process described above. In the example, we calculate expressions for $\Pr[w_1 = 0, w_2 = 1]_0$, $\Pr[w_1 = 1, w_2 = 0]_0$, $\Pr[w_1 = 0, w_2 = 1]_1$, $\Pr[w_1 = 1, w_2 = 0]_1$, $\Pr[w_1 = 0, w_2 = 2]_1$, $\Pr[w_1 = 1, w_2 = 1]_0$. We start with the construction of the balance equations for states $(0, 0)$ and $(0, 1)$, i.e.:

$$\begin{aligned}
\Pr[w_1 = 0, w_2 = 0] =& p_{00}\big(\Pr[w_1 = 1, w_2 = 0] + \Pr[w_1 = 0, w_2 = 1] \\
&+ \Pr[w_1 = 0, w_2 = 0]\big), \\
\Pr[w_1 = 0, w_2 = 1] =& p_{00}\big(\Pr[w_1 = 0, w_2 = 2] + \beta \Pr[w_1 = 1, w_2 = 1]\big) \\
&+ \frac{p_{01}}{p_{00}} \Pr[w_1 = 0, w_2 = 0].
\end{aligned}$$

$\Pr[w_1 = 0, w_2 = 0]$ is known and equals $1 - p_{01} - p_{10} - 2p_{11}$. Subsequently, we expand all probabilities as their power series in β, equate corresponding coefficients and solve the system of equations for the unknowns. This gives:

$$\begin{aligned}
\Pr[w_1 = 0, &w_2 = 1]_n = \frac{p_{01}}{p_{00}} \Pr[w_1 = 0, w_2 = 0]_n \\
&+ p_{00}\big(\Pr[w_1 = 0, w_2 = 2]_n + \Pr[w_1 = 1, w_2 = 1]_{n-1}\big), \\
\Pr[w_1 = 1, &w_2 = 0]_n = \frac{1 - p_{00} - p_{01}}{p_{00}} \Pr[w_1 = 0, w_2 = 0]_n \\
&- p_{00}\big(\Pr[w_1 = 0, w_2 = 2]_n + \Pr[w_1 = 1, w_2 = 1]_{n-1}\big),
\end{aligned}$$

where we also introduced the convention that $\Pr[w_1 = \cdot, w_2 = \cdot]_n$ with $n < 0$ equals zero. The constant coefficients are easily calculated by plugging in $n = 0$ as $\Pr[w_1 = 0, w_2 = 2]$ is of $\mathrm{O}(\beta)$ and $\Pr[w_1 = 1, w_2 = 1]_{-1} = 0$:

$$\begin{aligned}
\Pr[w_1 = 0, w_2 = 1]_0 =& \frac{p_{01}\big(1 - p_{01} - p_{10} - 2p_{11}\big)}{p_{00}}, \\
\Pr[w_1 = 1, w_2 = 0]_0 =& \frac{\big(1 - p_{00} - p_{01}\big)\big(1 - p_{01} - p_{10} - 2p_{11}\big)}{p_{00}}.
\end{aligned}$$

For the linear coefficients we have:

$$\begin{aligned}
\Pr[w_1 = 0, &w_2 = 1]_1 \\
&= p_{00}\Big(\Pr[w_1 = 0, w_2 = 2]_1 + \Pr[w_1 = 1, w_2 = 1]_0\Big), \\
\Pr[w_1 = 1, &w_2 = 0]_1 \\
&= -p_{00}\Big(\Pr[w_1 = 0, w_2 = 2]_1 + \Pr[w_1 = 1, w_2 = 1]_0\Big).
\end{aligned}$$

From the balance equation for state $(0, 2)$ we further get:

$$\Pr[w_1 = 0, w_2 = 2]_1 = \frac{p_{01}}{1 - p_{01}} \Pr[w_1 = 1, w_2 = 1]_0.$$

Lastly for $\Pr[w_1 = 1, w_2 = 1]_0$, from the balance equations for the second row (states $(1, 0)$ and $(1, 1)$), we expand:

$$\begin{aligned} & p_{00}\big(\Pr[w_1 = 2, w_2 = 0]_0 + \Pr[w_1 = 1, w_2 = 1]_0\big) \\ &\quad = \Pr[w_1 = 1, w_2 = 0]_0 + p_{10}\big(\Pr[w_1 = 0, w_2 = 0]_0 \\ &\qquad\qquad + \Pr[w_1 = 1, w_2 = 0]_0 + \Pr[w_1 = 0, w_2 = 1]_0\big), \\ & (1 - p_{01}) \Pr[w_1 = 1, w_2 = 1]_0 - p_{01} \Pr[w_1 = 2, w_2 = 0]_0 \\ &\quad = p_{11}\big(\Pr[w_1 = 0, w_2 = 0]_0 + \Pr[w_1 = 1, w_2 = 0]_0 \\ &\qquad\qquad + \Pr[w_1 = 0, w_2 = 1]_0\big). \end{aligned}$$

Solving this system of equations, we find:

$$\begin{aligned} \Pr[w_1 = 1, w_2 = 1]_0 = & \frac{p_{01}}{p_{00}} \Pr[w_1 = 1, w_2 = 0]_0 \\ & + \frac{p_{00}p_{11} - p_{01}p_{10}}{p_{00}^2} \Pr[w_1 = 0, w_2 = 0]_0. \end{aligned}$$

In this small example, we calculated expressions for $\Pr[w_1 = 0, w_2 = 1]_0$, $\Pr[w_1 = 1, w_2 = 0]_0$, $\Pr[w_1 = 0, w_2 = 1]_1$, $\Pr[w_1 = 1, w_2 = 0]_1$, $\Pr[w_1 = 0, w_2 = 2]_1$, $\Pr[w_1 = 1, w_2 = 1]_0$ that are only a function of the input parameters (and of each other). Continuing this pattern all coefficients of each state can be calculated. The result of that process are the expressions listed in (6.13)-(6.18). Each resulting expression contains only coefficients of state probabilities in such a way that in a finite number of successive substitutions an expression is obtained that is an exclusive function of the input parameters.

6.3 Summary of the algorithm and numerical complexity

In this section, we take a closer look at the algorithm resulting from the formulas. We derive the numerical complexity to estimate the scaling behavior of the algorithm. Furthermore, we list which measures need to be calculated to obtain the M-th coefficient of $\mathrm{E}[w_2]$.

To calculate $\mathrm{E}[w_2]_M$, we see from the theorems that all coefficients of lower order of $\mathrm{E}[w_2]$ are needed ($\mathrm{E}[w_2]_j : \forall j < M$). Furthermore, $\mathrm{E}[w_2 \mathbb{1}_{w_1=i}]_j$ and $\Pr[w_1 = i]_j$ for $j < M$ and $i < M - j$ are needed. During these calculations, numerous joint probability coefficients $\Pr[w_1 = i, w_2 = l]_n$ are required for which the calculation in turn requires more joint probability coefficients. For instance to calculate $\Pr[w_1 = 2, w_2 = 3]_4$ indicated by the solid disk in Figure 6.2, all coefficients indicated by circles, need to be calculated. This is done using the equations presented in (6.13)-(6.18).

It is thus clear that the complexity to calculate $\mathrm{E}[w_2]_M$ is of the same complexity as calculating the complete power series of $\mathrm{E}[w_2]$ up to the M-th coefficient. This calculation is summarized in Algorithm 6.1. The algorithm requires a cache to store the values calculated in the last two top-level iterations. For instance calculations in iteration j require values calculated in iterations $j-1$ and $j-2$. Calculating the coefficients in *ascending* order avoids the need to remember all previously calculated values and thus minimizes memory consumption. Lastly, from the three levels deep `for`-loop in Algorithm 6.1, we derive that the algorithm is of $\mathrm{O}(M^3)$.

```
Calculate $\mathrm{E}[w_2]_0$
for $i = 1$ to $M$ do
    /* calculate joint probabilities using
       (6.13)-(6.18)                                  */
    for $n = 0$ to $i - 1$ do
        for $l = 1$ to $n + 1$ do
            Calculate $\Pr[w_1 = i - n - 1, w_2 = l]_n$
        end for
    end for
    /* calculate probabilities using Theorem 6.3      */
    for $n = 0$ to $i - 1$ do
        Calculate $\Pr[w_1 = i - n - 1]_n$
    end for
    /* calculate partial means using Theorem 6.2      */
    for $n = 0$ to $i - 1$ do
        Calculate $\mathrm{E}[w_2 \mathbb{1}_{w_1 = i-n-1}]_n$
    end for
    Calculate $\mathrm{E}[w_2]_i$
end for
```

Algorithm 6.1: Algorithm to calculate the power series of $\mathrm{E}[w_2]$ up to the M-th coefficient.

6.4 Numerical continuation of the power series

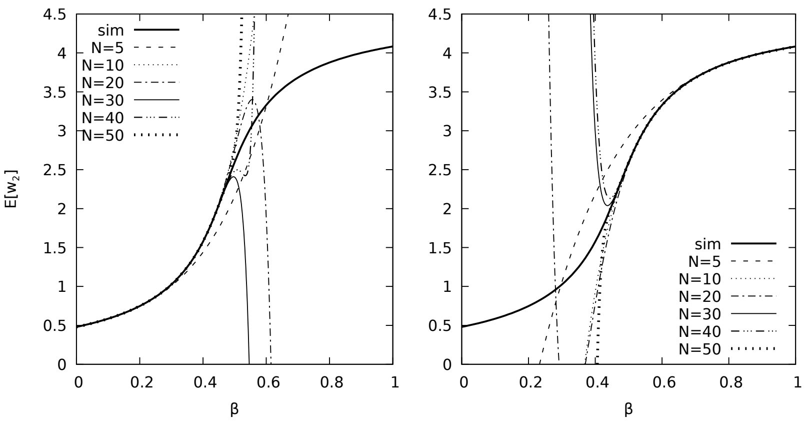
All techniques involving a Taylor series expansion at a certain value of one of the system parameters pose the same challenge of constructing a good approximation from it. The challenge is twofold [62,74,128]. A first part is deriving the radius of convergence of the power series. The power series of a given function is centered around a certain point and the radius of convergence of that series is the largest disk around that point for which the series converges and equals that given function[1]. A second part of the challenge is extending an approximation outside this convergence radius. In this section, we address both parts of this challenge.

One simple heuristic for determining the radius of convergence is by checking if the power series still converges by verifying if the result does not change significantly by for instance doubling the number of coefficients [65]. The radius of convergence is limited by the singularity *closest* to the expansion point. Deriving these singularities from the power series is non-trivial. Some methods are Domb-Sykes [53] for a single singularity on the real axis or Hunter and Guerrieri [77] for a pair of complex conjugate singularities. Other methods are also referenced in those two papers.

For our algorithm, we start by looking at the raw performance of the truncated power series. The results from sections 6.1 and 6.2 provide us with all necessary equations for the iterative calculation of the coefficients of the power series in β of $\mathrm{E}[w_2]$. In Figure 6.3a, we compare the truncated power series for $\mathrm{E}[w_2]$, i.e.: $\sum_{n=0}^{M} \mathrm{E}[w_2]_n \beta^n$, with simulation results for an example with arrival parameters $p_{00} = 0.46; p_{01} = 0.12; p_{10} = 0.06; p_{11} = 0.36$. These simulation results were obtained for 101 equally spaced values of β. From the graphs, we see that the approximations are extremely accurate for $\beta < 0.5$. For $\beta > 0.5$, we approach (and cross) the border of the convergence radius for the power series. The power series diverges in that region and no longer equals $\mathrm{E}[w_2]$; thus for higher M the performance of the power series deteriorates. As such the truncated power series does not provide good results for all values $\beta \in [0, 1]$.

A partial solution is to use the symmetry of the queueing system at hand, as was extensively discussed before. We define the tilde-system, which is the symmetric system, whereby class-1 customers are sent to

[1]Note that for the power series in Chapter 4 and Chapter 5 we could only calculate the first few coefficients. In those cases accurately determining the radius of convergence from the power series coefficients is impossible.



(a) Regular system (b) Symmetric system

Figure 6.3: Comparison of simulation with power series result.

queue 2 and vice-versa, i.e.: $\tilde{p}_{00} = p_{00}; \tilde{p}_{10} = p_{01}; \tilde{p}_{01} = p_{10}; \tilde{p}_{11} = p_{11}$. In this system $\mathrm{E}[\tilde{w}_1]$ for $\tilde{\beta} = 1 - \beta$ equals $\mathrm{E}[w_2]$ in the regular system. Via this symmetric system, we calculate the power series for $\mathrm{E}[w_2]$ in $\beta = 1$. This produces the result from Figure 6.3b, where we observe the same problem as in Figure 6.3a. We conclude that, in this example, the convergence radius of the power series of $\mathrm{E}[w_2]$ both in $\beta = 0$ and $\beta = 1$ is smaller than 1.

If $\mathrm{E}[w_2]$ is only required for a single value of β the easiest approximation is checking if the power series of either the regular or symmetric (tilde) system converges for this β value. This convergence is most easily tested by verifying the change of doubling the number of coefficients, as mentioned before. If β is however outside/very close to the radius of convergence of both systems or we require a single approximation for the full β spectrum, none of both truncated power series deliver satisfactory results.

To extend the approximation beyond the radius of convergence of the power series (also known as numerical continuation), we have already demonstrated the use of Padé approximants in the previous chapters. For higher M the Padé approximants have numerous unknowns and calculating them is computationally intensive (we were only able to calculate Padé approximants up to $N = 6$). Furthermore, it worsens the problems with local extrema and poles as described in Section 4.3.

Another method for numerical continuation is to find the exact location of the singularities that limit the convergence radius and use this knowledge explicitily to our advantage. In this example, the power series have convergence radii smaller than 1; as such there needs to be a singularity inside the unit circle. Given the nature of the function at hand, $\mathrm{E}[w_2]$ defined everywhere for $\beta \in [0, 1]$, a singularity in the real interval $[0, 1]$ is impossible. Assuming the singularity limiting the convergence radius is a conjugate, complex pair of singularities inside the unit circle, we estimate the position of these singularities from the power series. To this end, we use the method of Hunter and Guerrieri [77] together with the notes of Corliss [43]. The result of this method is an estimate of R and θ, the norm and argument of one of the singularities. Clearly, the complex conjugate has the same norm R which also equals the convergence radius of the power series and an opposite argument, $-\theta$. Furthermore, Hunter and Guerrieri estimate ν which is an index describing the nature of the singularity. We write for β in the neighborhood of R:

$$\mathrm{E}[w_2] \approx K(\beta - R \cdot e^{\imath\theta})^{\nu}(\beta - R \cdot e^{-\imath\theta})^{\nu},$$

with $\imath$ the complex unit and K a constant.

For the power series of $\mathrm{E}[w_2]$ in $\beta = 1$ (calculated via the symmetric case as described earlier), we can also search for the dominant singularities. This results in an R', θ' and ν'. In all examples we tested we found that $R = R'$, $\theta = \theta'$ and $\nu = \nu'$. This means that there are precisely 2 conjugate and complex singularities with $R \cos\theta < 1$. If more singularities existed with real part in $[0, 1]$ one of both ($\beta = 0$ and $\beta = 1$) power series would indeed yield a different dominant singularity. However, we were not able to prove only two conjugate dominant singularities exist for all possible cases. In the remainder, we assume this result is true in general, nonetheless the remainder is easily extended to incorporate 2 pairs of conjugate singularities.

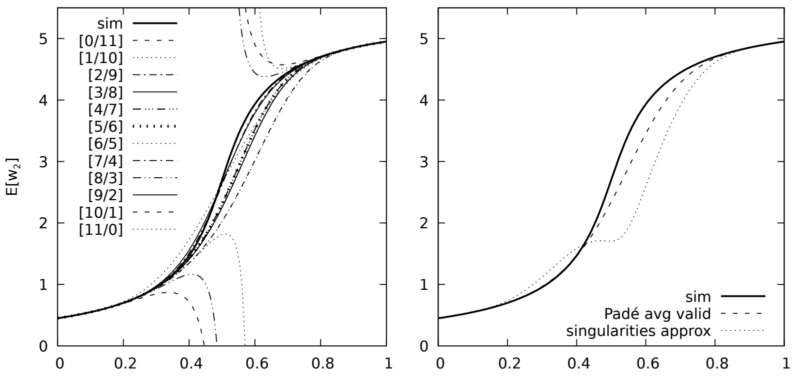
Subsequently, we construct an approximation method from these singularities. We reuse the principle of Padé approximants and enforce an approximation of rational form that (1) matches the first M coefficients of the power series in both endpoints and (2) has the correct singularities. As such, we obtain a new and simple approximation of the form:

$$\frac{\sum_{j=0}^{2M+1} c_j \beta^j}{(\beta - R \cdot e^{\imath\theta})^{-\nu}(\beta - R \cdot e^{-\imath\theta})^{-\nu}}. \tag{6.19}$$

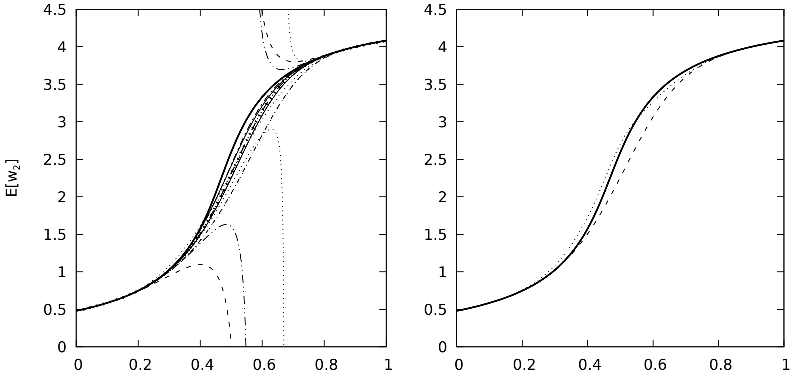
The denominator is calculated using the Hunter and Guerrieri method, the variables c_j in the numerator are calculated in such a way to match the values and first M derivatives of $\mathrm{E}[w_2]$ in both β equal to 0 and 1.

In Figure 6.4, we compare the approximations from Padé approximants and the approximant of the form of equation (6.19) with simulation results. The Padé approximants are shown on the left side and the titles in the legend use our normal nomenclature, i.e., [degree of numerator / degree of denominator]. The approximants match the first 6 coefficients of the power series in $\beta = 0$ and $\beta = 1$. The result of Algorithm 4.1 (i.e. filtering invalid approximants with poles or extrema and averaging the remaining approximants) is displayed on the right of Figure 6.4 together with the approximant of the form of equation (6.19). This last approximation includes the singularities found with the method of Hunter and Guerrieri's method (to find R, θ and ν) and is executed using the coefficients of the power series of $\mathrm{E}[w_2]$ from order 10 to order 50. Furthermore, M is chosen equal to 5, such that 6 coefficients are matched on both sides of the approximation (like for the Padé approximant).

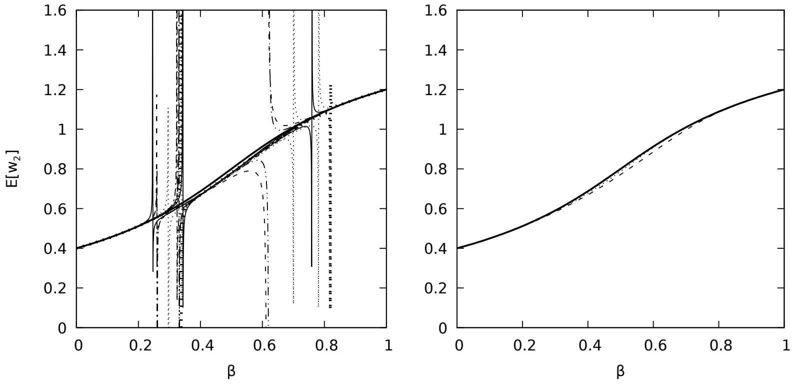
Clearly the result from the Padé approximants is better than in Section 4.4, as we are now able to calculate more coefficients in the power series (than via the generating function) and thus match more coefficients. Our newly constructed approximation method from equation (6.19) does not always perform better than the average of the valid Padé approximants, as for instance in Figure 6.4a. In fact, for that example both approximations on the right are not symmetric (point-symmetry for $\beta = 0.5$), which should be the case as the arrival process is completely symmetric ($p_{01} = p_{10}$). In those symmetric cases there is definitely room for improvement by enforcing this symmetry of the result in the construction of Padé approximants or the form of equation (6.19). For the examples in Figure 6.4b and 6.4c the approximation including the singularities in the approximation performs better than the Padé approximants, proving that this approach certainly has its value.



(a) $p_{00} = 0.55; p_{01} = p_{10} = 0; p_{11} = 0.45$



(b) $p_{00} = 0.46; p_{01} = 0.12; p_{10} = 0.06; p_{11} = 0.36$



(c) $p_{00} = 0.36; p_{01} = p_{10} = 0.24; p_{11} = 0.16$

Figure 6.4: Comparison of simulation with (left) Padé approximants matching 6 coefficients on both sides and (right) average of *valid* Padé approximants & approximation including the singularities of the form of equation (6.19) also matching 6 coefficients on both sides.

"De laatste koe doet de deur toe."
— Germaine Deryckere

7 Conclusions

In this dissertation, we studied a discrete-time, probabilistic implementation of Generalized Processor Sharing. This is a scheduler that can be used to schedule the service of several classes of customers with the goal to achieve service differentiation. This service differentiation, in turn, makes it possible to guarantee a sufficient Quality of Service for the different customer classes in the system. The analysis and optimization of GPS schedulers is known to be very difficult even for the most elementary arrival processes. Few analytical results are available and they are mostly limited to the boundary cases, where the GPS scheduler in fact degenerates to a priority scheduler.

First, we defined the two-class model. In this model, in case of contention, class-1 (class-2) is served with probability β $(1-\beta)$. In all other cases, the backlogged class (if any) is served. Furthermore, we considered a general arrival process that is identical and independent from slot to slot whereby every customer requires a single slot of service. In Chapter 2, we first proved two important properties about the performance measures of the system. We proved that (1) the system is work-conserving, i.e. the total amount of unfinished work in the system is independent of β and (2) that the mean unfinished work of class 2 (class 1) is a monotonically increasing (decreasing) function of β.

Using these important properties, we showed that it is possible to study the behavior and the optimum of any objective function that is a function of the mean unfinished work of both classes. As we proved that $\bar{w}_2$ and β are related through a bijection, this study is done in terms of $\bar{w}_2$. Furthermore, we showed that in some cases it is possible to rewrite an objective function that is a function of some other performance measure, like the mean delay, as an objective function in the mean unfinished work.

For a specific class of objective functions, namely a convex combination (with parameter γ) of functions g_j of the mean unfinished work of both classes, we studied the influence of the choice of g_j and γ on the optimum. We proved that if the g_j are linear or concave increasing functions (in the region of interest) that the optimum will always be in one of the endpoints, i.e. either $\beta = 0$ or $\beta = 1$ which correspond to priority cases. If the g_j are convex increasing functions, there is a single optimum in $\beta \in]0, 1[$ (pure GPS) when γ is chosen in a specific interval that can be calculated a priori in terms of the model parameters ($]\phi(0), \phi(1)[$). Furthermore, we proved some theorems that allow the same kind of analysis for general functions g_j. We also made the important side note that the analysis of this subclass of objective functions relies on few properties of the GPS scheduler. We showed that these properties are also fulfilled in several other settings and that the analysis thus also transfers to these situations. Specifically, we discussed the famous $c\mu$-rule and a semi-preemptive priority scheduler, in extenso.

The analysis of these objective functions gives us values for the extrema in terms of $\bar{w}_2$, for the configuration of the GPS system; however, we require the corresponding β-value. In Chapter 2, we studied and suggested some algorithms that are based on simulations of the system to find this optimal β-value. These algorithms are, once more, based on the important properties that we proved at the start of the chapter.

Another possibility to find the correspondence between $\bar{w}_2$ and β is the analysis of the system itself. In Chapter 4, we summarized a method from the literature that iteratively calculates higher-order coefficients in the power series of $\bar{w}_2$ in $\beta = 0$. This power series for the mean value is derived from a power series of the probability generating function. We discussed the challenges associated with this method, namely the fact that only a limited amount of coefficients can be calculated, due to computational limitations resulting from repeated use of l'Hôpital's rule for the conversion from the pgf. We proposed some solutions for this and studied their performance and accuracy in an optimization setting. Furthermore, we showed some possible modifications to the implementation of this method to maximize the number of calculable coefficients and thus the resulting performance.

Lastly, in Chapter 6, we derived the same power series for $\bar{w}_2$ but now directly from the system equations without going through the pgf. This enabled us to calculate more coefficients and determine the dominant singularities that limit the convergence radius of the power series. Using these coefficients and the singularities, we proposed a better approximation method. The discussion in this chapter was limited to a Bernoulli arrival process, to make the study less intricate as a starting point. The restriction to this arrival process, however, does not simplify the inherent difficulties associated with GPS schedulers which is the subject of this work.

In Chapter 3, we investigated the extension of two-class discrete-time GPS to three classes. We showed that a hierarchical implementation (H-GPS) has some important advantages compared to a non-hierarchical one. The most important one is that the optimization can also be done hierarchically, as β_2, the probability parameter on the second hierarchy level, has no influence on the customers on level 1 (queue 1). However, this single H-GPS system has a smaller performance region than a non-hierarchical GPS system; the latter can achieve any performance in the simplex span by the vertices of strict priority performances, while the former cannot. H-GPS, though, has three possible configurations by permuting which customer class is enqueued on the highest level in the hierarchy. We proved that the union of the performance regions of these three configurations coincides with the performance region of GPS. Furthermore, we used some important steps from this proof to construct an algorithm to fast and accurately select which configurations to use for a specific desired performance.

A second advantage of H-GPS is that we showed in Chapter 5, that we could extend the power series approximation for the two-class system, summarized in Chapter 4, to this three-class H-GPS system. The result is a two-dimensional power series centered around any of the boundary configuration points, i.e. $(\beta_1, \beta_2) = \{(0,0), (1,0), (0,1), (1,1)\}$. Furthermore, we studied the special case where one high-priority class has priority over two other classes that share the remaining capacity using GPS. In the numerical examples for this case, we thoroughly investigated the influence of the amount of high-priority customers on the other two classes.

Bibliography

[1] S Aalto, U Ayesta, SC Borst, V Misra, and R Núñez-Queija. Beyond processor sharing. *ACM SIGMETRICS Performance Evaluation Review*, 34(4):36–43, 2007.

[2] IJBF Adan, OJ Boxma, and JAC Resing. Queueing models with multiple waiting lines. *Queueing Systems*, 37(1-3):65–98, 2001.

[3] IJBF Adan, JSH Van Leeuwaarden, and Erik MM Winands. On the application of Rouché's theorem in queueing theory. *Operations Research Letters*, 34(3):355–360, 2006.

[4] IJBF Adan, J Wessels, and WHM Zijm. A compensation approach for two-dimensional Markov processes. *Advances in Applied Probability*, 25(4):783–817, 1993.

[5] E Altman, K Avrachenkov, and U Ayesta. A survey on discriminatory processor sharing. *Queueing Systems*, 53(1-2):53–63, 2006.

[6] E Altman and U Yechiali. Polling in a closed network. *Probability in the Engineering and Informational Sciences*, 8(3):327–343, 1994.

[7] S Asmussen. *Applied probability and queues*, volume 51. Springer Science & Business Media, 2008.

[8] S Asmussen and PW Glynn. *Stochastic simulation: algorithms and analysis*, volume 57. Springer Science & Business Media, 2007.

[9] K Avrachenkov and T Bodas. On the equivalence between multiclass processor sharing and random order scheduling policies. *ACM SIGMETRICS Performance Evaluation Review*, 45(4):2–6, 2018.

[10] KE Avrachenkov, JA Filar, and PG Howlett. *Analytic perturbation theory and its applications.* SIAM, 2013.

[11] U Ayesta. A unifying conservation law for single-server queues. *Journal of Applied Probability*, 44(4):1078–1087, 2007.

[12] U Ayesta, A Izagirre, and IM Verloop. Heavy traffic analysis of the discriminatory random-order-of-service discipline. *ACM SIGMETRICS Performance Evaluation Review*, 39(2):41–43, 2011.

[13] JS Baras, DJ Ma, and AM Makowski. K competing queues with geometric service requirements and linear costs: The μc-rule is always optimal. *Systems & Control Letters*, 6(3):173–180, 1985.

[14] JCR Bennett, DC Stephens, and H Zhang. High speed, scalable, and accurate implementation of packet fair queueing algorithms in ATM networks. In *Network Protocols, 1997. Proceedings., 1997 International Conference on*, pages 7–14. IEEE, 1997.

[15] JCR Bennett and H Zhang. Hierarchical packet fair queueing algorithms. *IEEE/ACM Transactions on networking*, 5(5):675–689, 1997.

[16] DP Bertsekas. *Nonlinear programming.* Athena scientific Belmont, 1999.

[17] D Bertsimas. The achievable region method in the optimal control of queueing systems; formulations, bounds and policies. *Queueing Systems*, 21(3-4):337–389, 1995.

[18] D Bertsimas and J Niño-Mora. Conservation laws, extended polymatroids and multiarmed bandit problems; a polyhedral approach to indexable systems. *Mathematics of Operations Research*, 21(2):257–306, 1996.

[19] D Bertsimas and J Niño-Mora. Optimization of multiclass queueing networks with changeover times via the achievable region approach: Part i, the single-station case. *Mathematics of Operations Research*, 24(2):306–330, 1999.

[20] D Bertsimas, IC Paschalidis, and JN Tsitsiklis. Large deviations analysis of the generalized processor sharing policy. *Queueing Systems*, 32(4):319–349, 1999.

[21] D Bertsimas, ICH Paschalidis, and JN Tsitsiklis. Optimization of multiclass queueing networks: Polyhedral and nonlinear characterizations of achievable performance. *The Annals of Applied Probability*, pages 43–75, 1994.

[22] CF Bispo. The single-server scheduling problem with convex costs. *Queueing Systems*, 73(3):261–294, 2013.

[23] JPC Blanc. On a numerical method for calculating state probabilities for queueing systems with more than one waiting line. *Journal of Computational and Applied Mathematics*, 20:119–125, 1987.

[24] JPC Blanc. A numerical study of a coupled processor model. In *Computer performance and reliability*, volume 2, pages 289–303, 1988.

[25] B Blaszczyszyn, T Rolski, and V Schmidt. Light-traffic approximations in queues and related stochastic models. *Advances in Queueing, CRC Press, Boca Raton*, pages 379–406, 1995.

[26] SC Borst, OJ Boxma, and P Jelenkovic. Asymptotic behavior of generalized processor sharing with long-tailed traffic sources. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 912–921. IEEE, 2000.

[27] SC Borst, OJ Boxma, and P Jelenković. Reduced-load equivalence and induced burstiness in GPS queues with long-tailed traffic flows. *Queueing Systems*, 43(4):273–306, 2003.

[28] OJ Boxma. *Analysis and optimization of polling systems.* CWI Amsterdam, 1991.

[29] OJ Boxma. Static optimization of queueing systems. In *Recent Trends in Optimization Theory and Applications*, pages 1–16. World Scientific, 1995.

[30] OJ Boxma and JA Weststrate. Waiting times in polling systems with Markovian server routing. In *Messung, Modellierung und Bewertung von Rechensystemen und Netzen*, pages 89–104. Springer, 1989.

[31] RP Brent. *Algorithms for minimization without derivatives.* Courier Corporation, 2013.

[32] H Bruneel and BG Kim. *Discrete-time models for communication systems including ATM*, volume 205. Springer Science & Business Media, 2012.

[33] C Buyukkoc, P Variaya, and J Walrand. cμ rule revisited. *Adv. Appl. Prob.*, 17(1):237–238, 1985.

[34] HM Chaskar and U Madhow. Fair scheduling with tunable latency: a round-robin approach. *IEEE/ACM Transactions on Networking (TON)*, 11(4):592–601, 2003.

[35] MX Chen and SH Liu. Hierarchical deficit round-robin packet scheduling algorithm. In *Advances in Intelligent Systems and Applications-Volume 1*, pages 419–427. Springer, 2013.

[36] Cisco. VoIP over PPP links with quality of service. published electronically at `https://www.cisco.com/c/en/us/support/docs/voice/voice-quality/7111-voip-mlppp.html`, 2018.

[37] D Claeys. *Analysis of queueing models with batch service.* PhD thesis, Ghent University, 2011.

[38] D Claeys, K Laevens, J Walraevens, and H Bruneel. Complete characterisation of the customer delay in a queueing system with batch arrivals and batch service. *Mathematical Methods of Operations Research*, 72(1):1–23, Aug 2010.

[39] D Claeys, J Walraevens, K Laevens, and H Bruneel. Analysis of threshold-based batch-service queueing systems with batch arrivals and general service times. *Performance Evaluation*, 68(6):528 – 549, 2011.

[40] JW Cohen. Boundary value problems in queueing theory. *Queueing Systems*, 3(2):97–128, 1988.

[41] JW Cohen and OJ Boxma. *Boundary value problems in queueing system analysis*, volume 79. Elsevier, 2000.

[42] AR Conn, K Scheinberg, and LN Vicente. *Introduction to derivative-free optimization.* SIAM, 2009.

[43] G Corliss. On computing Darboux type series analyses. *Nonlinear Analysis: Theory, Methods & Applications*, 7(11):1247–1253, 1983.

[44] DR Cox and WL Smith. *Queues*, volume 2. CRC Press, 1961.

[45] M Dacre, K Glazebrook, and J Niño-Mora. The achievable region approach to the optimal control of stochastic systems. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, pages 747–791, 1999.

[46] E De Cuypere. *Numerical methods for queues with shared service.* PhD thesis, Ghent University, 2014.

[47] K De Turck, E De Cuypere, S Wittevrongel, and D Fiems. Algorithmic approach to series expansions around transient Markov chains with applications to paired queuing systems. In *Performance Evaluation Methodologies and Tools (VALUE-TOOLS), 2012 6th International Conference on*, pages 38–44. IEEE, 2012.

[48] G de Veciana, G Kesidis, and J Walrand. Resource management in wide-area ATM networks using effective bandwidths. *IEEE Journal on Selected Areas in Communications*, 13(6):1081–1090, 1995.

[49] G De Veciana and J Walrand. Effective bandwidths: Call admission, traffic policing and filtering for ATM networks. *Queueing Systems*, 20(1-2):37–59, 1995.

[50] K Debicki and M Mandjes. A note on large-buffer asymptotics for generalized processor sharing with Gaussian inputs. *Queueing Systems*, 55(4):251–254, 2007.

[51] T Demoor. *Priority queues with limited capacity.* PhD thesis, Ghent University, 2014.

[52] JE Dennis and VJ Torczon. Derivative-free pattern search methods for multidisciplinary design problems. In *The Fifth AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, pages 922–932, 1994.

[53] C Domb and MF Sykes. On the susceptibility of a ferromagnetic above the Curie point. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 240, pages 214–228. The Royal Society, 1957.

[54] AN Dudin, MH Lee, O Dudina, and SK Lee. Analysis of priority retrial queue with many types of customers and servers reservation as a model of cognitive radio system. *IEEE Transactions on Communications*, 65(1):186–199, 2017.

[55] A Elwalid and D Mitro. Design of generalized processor sharing schedulers which statistically multiplex heterogeneous QoS classes. In *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1220–1230. IEEE, 1999.

[56] AK Erlang. Solutions of some problems in the theory of probabilities of significance in automatic telephone exchanges. 13:5–13, 01 1917.

[57] E Evdokimova, S Wittevrongel, and D Fiems. A Taylor series approach for service-coupled queueing systems with intermediate load. *Mathematical Problems in Engineering*, 2017, 2017.

[58] G Fayolle and R Iasnogorodski. Two coupled processors: the reduction to a Riemann-Hilbert problem. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 47(3):325–351, 1979.

[59] G Fayolle, I Mitrani, and R Iasnogorodski. Sharing a processor among many job classes. *Journal of the ACM (JACM)*, 27(3):519–532, 1980.

[60] A Federgruen and H Groenevelt. Characterization and optimization of achievable performance in general queueing systems. *Operations Research*, 36(5):733–741, 1988.

[61] A Federgruen and H Groenevelt. M/G/c queueing systems with multiple customer classes: characterization and control of achievable performance under nonpreemptive priority rules. *Management Science*, 34(9):1121–1138, 1988.

[62] AV Ferris-Prabhu and DH Withers. Numerical analytic continuation using Padé approximants. *Journal of Computational Physics*, 13(1):94–99, 1973.

[63] M Fiedler, T Hossfeld, and P Tran-Gia. A generic quantitative relationship between quality of experience and quality of service. *Network, IEEE*, 24(2):36–41, 2010.

[64] D Fiems. *Analysis of discrete-time queueing systems with vacations.* PhD thesis, Ghent University, 2004.

[65] D Fiems and K De Turck. A series expansion approach for finite-capacity processor sharing queues. In *Proceedings of the 7th International Conference on Performance Evaluation Methodologies and Tools*, pages 118–125. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013.

[66] S Floyd and V Jacobson. Link-sharing and resource management models for packet networks. *IEEE/ACM transactions on Networking*, 3(4):365–386, 1995.

[67] E Gelenbe, X Mang, and R Onvural. Bandwidth allocation and call admission control in high-speed networks. *IEEE Communications Magazine*, 35(5):122–129, 1997.

[68] L Green. A queueing system with general-use and limited-use servers. *Operations Research*, 33(1):168–182, 1985.

[69] F Guillemin and D Pinchon. Analysis of generalized processor-sharing systems with two classes of customers and exponential services. *Journal of Applied Probability*, 41(3):832–858, 2004.

[70] MK Gupta, N Hemachandra, and J Venkateswaran. On completeness and equivalence of some dynamic priority schemes. Technical report, Tech. rep., IIT Bombay, 2014.

[71] JM Harrison. Dynamic scheduling of a multiclass queue: Discount optimality. *Operations Research*, 23(2):270–282, 1975.

[72] R Hassin, J Puerto, and FR Fernández. The use of relative priorities in optimizing the performance of a queueing system. *European Journal of Operational Research*, 193(2):476–483, 2009.

[73] MT Heath. *Scientific computing*. McGraw-Hill New York, 2002.

[74] P Henrici. An algorithm for analytic continuation. *SIAM Journal on Numerical Analysis*, 3(1):67–78, 1966.

[75] MF Homg, WT Lee, KR Lee, and YH Kuo. An adaptive approach to weighted fair queue with qos enhanced on ip network. In *TENCON 2001. Proceedings of IEEE Region 10 International Conference on Electrical and Electronic Technology*, volume 1, pages 181–186. IEEE, 2001.

[76] G Hooghiemstra, M Keane, and S Van de Ree. Power series for stationary distributions of coupled processor models. *SIAM Journal on Applied Mathematics*, 48(5):1159–1166, 1988.

[77] C Hunter and B Guerrieri. Deducing the properties of singularities of functions from their Taylor series coefficients. *SIAM Journal on Applied Mathematics*, 39(2):248–263, 1980.

[78] A Itai and Z Rosberg. A golden ratio control policy for a multiple-access channel. *IEEE Transactions on Automatic Control*, 29(8):712–718, 1984.

[79] X Jin and G Min. Analytical modelling of hybrid PQ-GPS scheduling systems under long-range dependent traffic. In *Advanced Information Networking and Applications, 2007. AINA'07. 21st International Conference on*, pages 1006–1013. IEEE, 2007.

[80] X Jin and G Min. Performance modelling of hybrid PQ-GPS systems under long-range dependent network traffic. *Communications Letters, IEEE*, 11(5):446–448, 2007.

[81] C Joo and NB Shroff. A novel coupled queueing model to control traffic via QoS-aware collision pricing in cognitive radio networks. In *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*, pages 1–9. IEEE, 2017.

[82] CR Kalmanek, H Kanakia, and S Keshav. Rate controlled servers for very high-speed networks. In *Global Telecommunications Conference, 1990, and Exhibition.'Communications: Connecting the Future', GLOBECOM'90., IEEE*, pages 12–20. IEEE, 1990.

[83] M Katevenis, S Sidiropoulos, and C Courcoubetis. Weighted round-robin cell multiplexing in a general-purpose ATM switch chip. *IEEE Journal on selected Areas in Communications*, 9(8):1265–1279, 1991.

[84] JC Ke, CH Wu, and ZG Zhang. Recent developments in vacation queueing models: a short survey. *International Journal of Operations Research*, 7(4):3–8, 2010.

[85] N Khedun and V Bassoo. Analysis of priority queueing with multichannel in cognitive radio network. In *EUROCON 2015-International Conference on Computer as a Tool (EUROCON), IEEE*, pages 1–6. IEEE, 2015.

[86] J Kim, B Kim, and H Luh. Analysis of a Markovian feedback queue with multi-class customers and its application to the weighted round-robin queue. *Annals of Operations Research*, pages 1–23, 2018.

[87] K Kim and KC Chae. Discrete-time queues with discretionary priorities. *European Journal of Operational Research*, 200(2):473–485, 2010.

[88] JFC Kingman. Two similar queues in parallel. *The Annals of Mathematical Statistics*, 32(4):1314–1323, 1961.

[89] L Kleinrock. *Queueing systems, volume 2: Computer applications*, volume 66. wiley New York, 1976.

[90] L Kleinrock and H Levy. The analysis of random polling systems. *Operations Research*, 36(5):716–732, 1988.

[91] IN Kovalenko. Rare events in queueing systems—a survey. *Queueing Systems*, 16(1):1–49, 1994.

[92] DP Kroese and V Schmidt. Light-traffic analysis for queues with spatially distributed arrivals. *Mathematics of operations research*, 21(1):135–157, 1996.

[93] K Kumaran, GE Margrave, D Mitra, and KR Stanley. Novel techniques for the design and control of generalized processor sharing schedulers for multiple QoS classes. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 932–941. IEEE, 2000.

[94] H Kushner and GG Yin. *Stochastic approximation and recursive algorithms and applications*, volume 35. Springer Science & Business Media, 2003.

[95] K Lange. *Numerical analysis for statisticians.* Springer Science & Business Media, 2010.

[96] HW Lee, C Kim, and S Chong. Scheduling and source control with average queue-length control in cellular networks. In *Communications, 2007. ICC'07. IEEE International Conference on*, pages 109–114. IEEE, 2007.

[97] M Lelarge. Asymptotic behavior of generalized processor sharing queues under subexponential assumptions. *Queueing Systems*, 62(1-2):51–73, 2009.

[98] C Li and MJ Neely. Solving convex optimization with side constraints in a multi-class queue by adaptive cμ rule. *Queueing Systems: Theory and Applications*, 77(3):331–372, 2014.

[99] P Lieshout. *Queueing models for bandwidth-sharing disciplines.* PhD thesis, Universiteit van Amsterdam, 2008.

[100] P Lieshout and M Mandjes. Generalized processor sharing: Characterization of the admissible region and selection of optimal weights. *Computers & Operations Research*, 35(8):2497–2519, 2008.

[101] P Lieshout, M Mandjes, and S Borst. GPS scheduling: selection of optimal weights and comparison with strict priorities. In *ACM SIGMETRICS Performance Evaluation Review*, volume 34, pages 75–86. ACM, 2006.

[102] T Maertens. *Analysis of discrete-time queueing systems with priority jumps*. PhD thesis, Springer, 2010.

[103] T Maertens, J Walraevens, and H Bruneel. A modified hol priority scheduling discipline: performance analysis. *European Journal of Operational Research*, 180(3):1168–1185, 2007.

[104] W Mélange, J Walraevens, D Claeys, B Steyaert, and H Bruneel. The impact of a global FCFS service discipline in a two-class queue with dedicated servers. *Computers & Operations Research*, 71:23–33, 2016.

[105] JA Morrison and SC Borst. Interacting queues in heavy traffic. *Queueing Systems*, 65(2):135–156, 2010.

[106] MJ Neely. Delay-based network utility maximization. *IEEE/ACM Transactions on Networking (TON)*, 21(1):41–54, 2013.

[107] JA Nelder and R Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[108] K Nichols, S Blake, F Baker, and D Black. Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers. RFC 2474 (Proposed Standard), dec 1998. Updated by RFCs 3168, 3260.

[109] AK Parekh and RG Gallager. A generalized processor sharing approach to flow control in integrated services networks-the multiple node case. In *INFOCOM'93. Proceedings. Twelfth Annual Joint Conference of the IEEE Computer and Communications Societies. Networking: Foundation for the Future, IEEE*, pages 521–530. IEEE, 1993.

[110] AK Parekh and RG Gallager. A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *IEEE/ACM transactions on networking*, 1(3):344–357, 1993.

[111] AS Parveen. A survey of an integrated scheduling scheme with long-range and short-range dependent traffic. *International Journal of Engineering Sciences & Research Technology*, 3(1):430–439, 2014.

[112] JM Pavlin. Dual bounds of a service level assignment problem with applications to efficient pricing. *European Journal of Operational Research*, 262(1):239–250, 2017.

[113] H Phan, TMC Chu, HJ Zepernick, and P Arlos. Packet loss priority of cognitive radio networks with partial buffer sharing. In *Communications (ICC), 2015 IEEE International Conference on*, pages 7646–7652. IEEE, 2015.

[114] WH Press, SA Teukolsky, WT Vetterling, and BP Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 3 edition, 2007.

[115] FL Presti, ZL Zhang, and D Towsley. Bounds, approximations and applications for a two-queue GPS system. In *IEEE INFOCOM*, volume 96, pages 1310–1317. INSTITUTE OF ELECTRICAL ENGINEERS INC (IEEE), 1996.

[116] RL Rardin. *Optimization in operations research*. Prentice Hall, 2016.

[117] MI Reiman and B Simon. Open queueing systems in light traffic. *Mathematics of operations research*, 14(1):26–59, 1989.

[118] B Rengarajan, C Caramanis, and G De Veciana. Analyzing queueing systems with coupled processors through semidefinite programming. *INFORMS: Applied Probability Session*, 2008.

[119] C Semeria. Supporting differentiated service classes: queue scheduling disciplines. *Juniper networks*, pages 11–14, 2001.

[120] JG Shanthikumar and DD Yao. Multiclass queueing systems: Polymatroidal structure and optimal scheduling control. *Operations Research*, 40(3-supplement-2):S293–S299, 1992.

[121] M Shreedhar and G Varghese. Efficient fair queueing using deficit round robin. In *ACM SIGCOMM Computer Communication Review*, volume 25, pages 231–242. ACM, 1995.

[122] RA Shumsky. Approximation and analysis of a call center with flexible and specialized servers. *OR Spectrum*, 26(3):307–330, 2004.

[123] NJA Sloane. A001147 - the on-line encyclopedia of integer sequences. published electronically at `https://oeis.org/A001147`, 2018.

[124] JC Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control*, volume 65. John Wiley & Sons, 2005.

[125] H Takagi. Queuing analysis of polling models. *ACM Computing Surveys (CSUR)*, 20(1):5–28, 1988.

[126] WB van den Hout. The power-series algorithm. a numerical approach to Markov processes. Technical report, Tilburg University, School of Economics and Management, 1996.

[127] RD van der Mei. *Polling systems and the power-series algorithm*. PhD thesis, Tilburg University, 1995.

[128] M Van Dyke. Analysis and improvement of perturbation series. *The Quarterly Journal of Mechanics and Applied Mathematics*, 27(4):423–450, 1974.

[129] G van Kessel, R Núñez-Queija, and SC Borst. Asymptotic regimes and approximations for discriminatory processor sharing. *SIGMETRICS Performance Evaluation Review*, 32(2):44–46, 2004.

[130] A Van Moorsel. Metrics for the internet age: Quality of experience and quality of business. In *Fifth International Workshop on Performability Modeling of Computer and Communication Systems, Arbeitsberichte des Instituts für Informatik, Universität Erlangen-Nürnberg, Germany*, volume 34, pages 26–31. Citeseer, 2001.

[131] M van Uitert. *Generalized processor sharing queues*. PhD thesis, Eindhoven University of Technology, 2003.

[132] RJ Vanderbei. *Linear programming*. Springer, 2015.

[133] J Vanlerberghe, T Maertens, J Walraevens, S De Vuyst, and H Bruneel. A hybrid analytical/simulation optimization of generalized processor sharing. In *Teletraffic Congress (ITC), 2013 25th International*, pages 1–9. IEEE, 2013.

[134] J Vanlerberghe, T Maertens, J Walraevens, S De Vuyst, and H Bruneel. On the optimization of two-class work-conserving parameterized scheduling policies. *4OR*, 14(3):281–308, 2016.

[135] J Vanlerberghe, J Walraevens, T Maertens, and H Bruneel. Approximating the optimal weights for discrete-time generalized processor sharing. In *Networking Conference, 2014 IFIP*, pages 1–9. IEEE, 2014.

[136] J Vanlerberghe, J Walraevens, T Maertens, and H Bruneel. On the influence of high priority customers on a generalized processor sharing queue. In *International Conference on Analytical and Stochastic Modeling Techniques and Applications*, pages 203–216. Springer, 2015.

[137] J Vanlerberghe, J Walraevens, T Maertens, and H Bruneel. Calculation of the performance region of an easy-to-optimize alternative for generalized processor sharing. *European Journal of Operational Research*, 270(2):625 – 635, 2018.

[138] J Vanlerberghe, J Walraevens, T Maertens, S De Vuyst, and H Bruneel. On generalized processor sharing and objective functions: Analytical framework. In *European Workshop on Performance Engineering*, pages 96–111. Springer, 2015.

[139] IM Verloop, U Ayesta, and S Borst. Monotonicity properties for multi-class queueing systems. *Discrete Event Dynamic Systems*, 20(4):473–509, 2010.

[140] VM Vishnevskii and OV Semenova. Mathematical methods to study the polling systems. *Automation and Remote Control*, 67(2):173–220, 2006.

[141] J Walraevens. *Discrete-time queueing models with priorities.* PhD thesis, Ghent University, 2004.

[142] J Walraevens, T Maertens, and H Bruneel. A semi-preemptive priority scheduling discipline: Performance analysis. *European Journal of Operational Research*, 224(2):324–332, 2013.

[143] J Walraevens, B Steyaert, and H Bruneel. Performance analysis of a single-server ATM queue with a priority scheduling. *Computers & Operations Research*, 30(12):1807–1829, 2003.

[144] J Walraevens, B Steyaert, M Moeneclaey, and H Bruneel. A discrete-time HOL priority queue with multiple traffic classes. In *International Conference on Networking*, pages 620–627. Springer, 2005.

[145] J Walraevens, JSH van Leeuwaarden, and OJ Boxma. Power series approximations for two-class generalized processor sharing systems. *Queueing Systems*, 66(2):107–130, 2010.

[146] J Walraevens, J Vanlerberghe, T Maertens, S De Vuyst, and H Bruneel. Strict monotonicity and continuity of mean unfinished work in two queues sharing a server. *Operations Research Letters*, 45(2):151–153, 2017.

[147] J Walraevens, S Wittevrongel, and H Bruneel. A discrete-time priority queue with train arrivals. *Stochastic Models*, 23(3):489–512, 2007.

[148] L Wang, G Min, DD Kouvatsos, and X Jin. Analytical modeling of an integrated priority and WFQ scheduling scheme in multi-service networks. *Computer Communications*, 33:S93–S101, 2010.

[149] L Xia and XR Cao. Performance optimization of queueing systems with perturbation realization. *European journal of operational research*, 218(2):293–304, 2012.

[150] O Yaron and M Sidi. Generalized processor sharing networks with exponentially bounded burstiness arrivals. *Journal of High Speed Networks*, 3(4):375–387, 1994.

[151] ZL Zhang. Large deviations and the generalized processor sharing scheduling for a two-queue system. *Queueing Systems*, 26(3-4):229–254, 1997.

[152] ZL Zhang. Large deviations and the generalized processor sharing scheduling for a multiple-queue system. *Queueing Systems*, 28(4):349–376, 1998.

[153] ZL Zhang, Z Liu, J Kurose, and D Towsley. Call admission control schemes under generalized processor sharing scheduling. *Telecommunication Systems*, 7(1-3):125–152, 1997.

[154] ZL Zhang, D Towsley, and J Kurose. Statistical analysis of generalized processor sharing scheduling discipline. *ACM SIGCOMM Computer Communication Review*, 24(4):68–77, 1994.