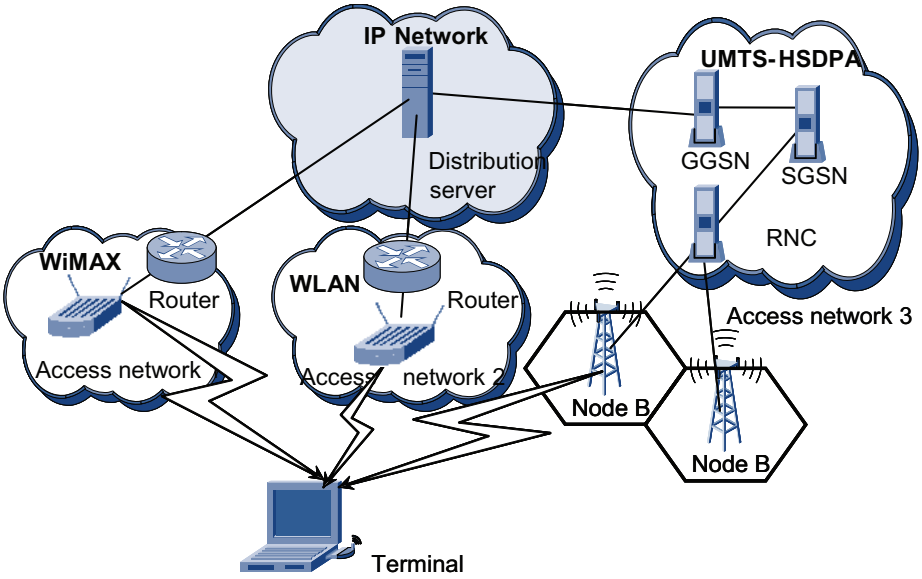
Concurrent Access in Wireless LANs



Author: Joost Bosman

2009

CWI
Centrum Wiskunde & Informatica

vrije Universiteit amsterdam

Supervisors CWI:
Rob van der Mei,
Gerard Hoekstra

Supervisor VU:
Sandjai Bhulai

Preface

The end of the BMI (Business Informatics and Mathematics) Master program consists of an internship that has to be carried out within a business, industry or research facility outside the Faculty of Exact Sciences of the VU. My Master project was at the CWI at the PNA2 (Probability and Stochastic Networks) research group. I am glad to announce that after my internship I will continue working as a PhD-student at the CWI.

I would like to thank prof.dr. Rob van der Mei for offering the opportunity for this internship. He was my internship supervisor at the CWI. His enthousiasm motivated me continue working on different ideas. Further I would like to thank dr. Sandjai Bhulai for being my supervisor at the VU. He provided me with good ideas and comments during my internship and on writing my thesis. I would like to thank ir. Gerard Hoekstra for his enthousiasm and support during the internship period. My internship was part of the Casimir project where Gerard is doing his PhD research. I would like to thank Sindo Nuñez-Queija for his input for my research on conditional sojourn time. Further I would like to thank my office mates Chrétien Verhoef and Arnoud den Boer. We had a really nice time in our room with the 'Cup a soup carroussel' and lots of strange Chinese teas and the adventures for finding the heat source that made our room really warm (even in winter). Finally I would like to thank my other colleagues at the CWI for the nice talks during lunch and coffee breaks.

Joost Bosman

September 2009

Contents

1 Introduction

1.1 About the CWI

Founded in 1946, CWI is the national research center for Mathematics and Computer Science in the Netherlands. More than 170 full professors have come from CWI, of whom 120 still are active. CWI's strength is the discovery and development of new ideas, and the transfer of knowledge to academia and to Dutch and European industry. This results in importance for our economy, from payment systems and cryptography to telecommunication and the stock market, from public transport and internet to water management and meteorology. With its 55 permanent research staff, 40 postdocs and 65 PhD students, CWI lies at the heart of European research in mathematics and computer science. Researchers at CWI are able to fully concentrate their efforts on their scientific work, and to build an international network of peers. More than half of the permanent research staff maintains close contact with universities as part-time professors. The personal and institutional research networks strengthen CWI's positions and serve as a magnet for attracting talent. CWI researchers come from more than 25 countries world-wide. A source of pride: CWI was a birthplace of the world-wide internet. Cwi.nl was the first national domain name ever issued anywhere. CWI helped develop the wing of the Fokker Friendship - chosen as the most beautiful Dutch design of the 20th century. The popular language Python was invented at CWI, the language in which Google was developed. CWI applied combinatorial algorithms to the scheduling of the Dutch railway system. XML-databases were build to the needs of the Netherlands Forensic Institute and 3D visualization techniques to better detect cancer tumors.

1.2 Summary

Today, we have a large number of wireless access networks at our disposal, ranging from hot-spot based local-area networks (e.g., Wireless LAN, WiMAX) to wide-area mobile networks, such as UMTS and HSDPA. This opens up the opportunity to enhance the user-perceived performance by using multiple access networks simultaneously, and to make applications more robust against the ever-changing circumstances in wireless access networks. The objective of this internship was to find optimal dynamical strategies for concurrent access in wireless LANs and to determine the impact of burstiness of the arriving traffic on the optimal strategies. These dynamical strategies are based on the observed number of flows which are the observed number of file transmissions on the wireless nodes. For this purpose MDPs (Markov Decision Processes) have been formulated. The MDPs can contain more states than the observable number of flows. Therefore a partial observation approach have been developed based that conditions the non observable states on the observed number of flows. For the partial observation problem also Bayesian dynamic programming models have been developed. These models use information states that represents the actual belief on the real state of the nodes. The information states have been combined with the full state observable MDP strategy. Further a simple decision heuristic has been developed that is based on the known result of the conditional sojourn time in a single PS (Processor Sharing) node. Finally all the strategies have been implemented in OPNET, a realistic network simulation environment, and the performance in terms of expected sojourn time has been compared. In this comparison the decision heuristic based on conditional sojourn time performs very well.

1.3 Motivation for Concurrent Access

Today, we have a large number of wireless access networks at our disposal, ranging from hot-spot based local-area networks (e.g., Wireless LAN, WiMAX) to wide-area mobile networks, such as UMTS and HSDPA. As a consequence, country-wide coverage is provided by multiple overlapping networks, especially in densely populated areas such as the Randstad area in the Netherlands. This opens up the opportunity to enhance the user-perceived performance by using multiple access networks simultaneously, and to make applications more robust against the ever-changing circumstances in wireless access networks. A new technology that enables the simultaneous use of concurrent networks is called Concurrent Access (CA). This creates a tremendous potential for application performance improvement. This performance inprovement can be realized in terms of

- Speed, when the bandwidth of two or more networks is combined a throughput can be realized higher than the maximum bandwidth of the separate networks. The improvement can be noticed in throughput but also in terms of access time. When multiple networks are used for a request the network with the fastest response can be used in order to archieve lower access times.

- Reliability, when two or more networks are used for downloading a connection loss does not nesessary stop the download. This can be compared with a download manager. A download manager uses multiple servers for downloading a file. If one of these servers does not repond anymore the download can still be completed using the other servers. The multiple download servers used in the download manager correspond multiple connections used with CA.

Figure 1 illustrates the use of multiple technologies for concurrent access.

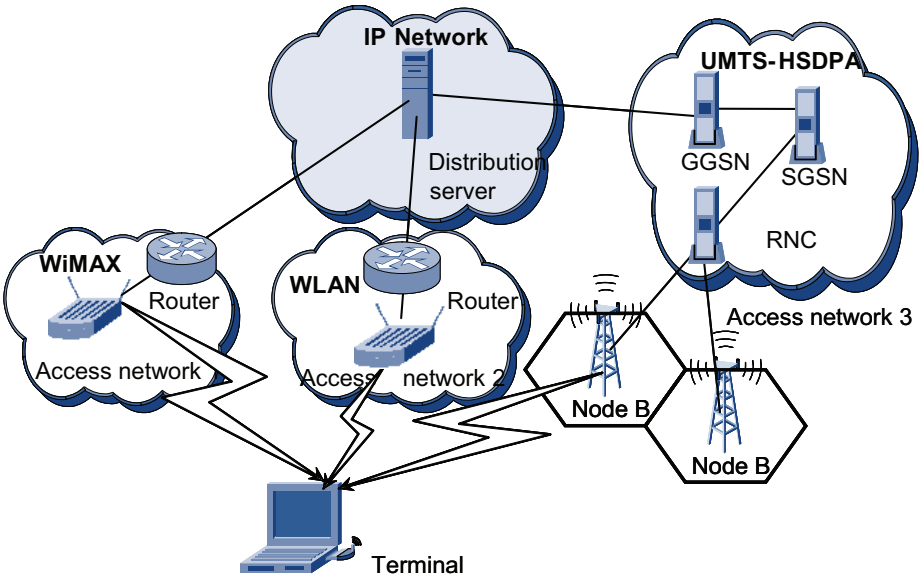


Figure 1: Concurrent access over multiple wireless technologies.

In Figure 1 the white clouds represent a variety of wireless technologies. The blue cloud represents a network that contains an application server from which files are requested. The terminal (displayed as a notebook computer) has concurrent access to the IP network with multiple connections. The concurrent access can be applied on connections using different wireless technologies or multiple connections that are using the same technology. In Figure 1 the UMTS-HSPDA connections to node A and node B are in the same wireless technology. The application of concurrent access is not limited to civil use. In Figure 2 it is applied in a military context.

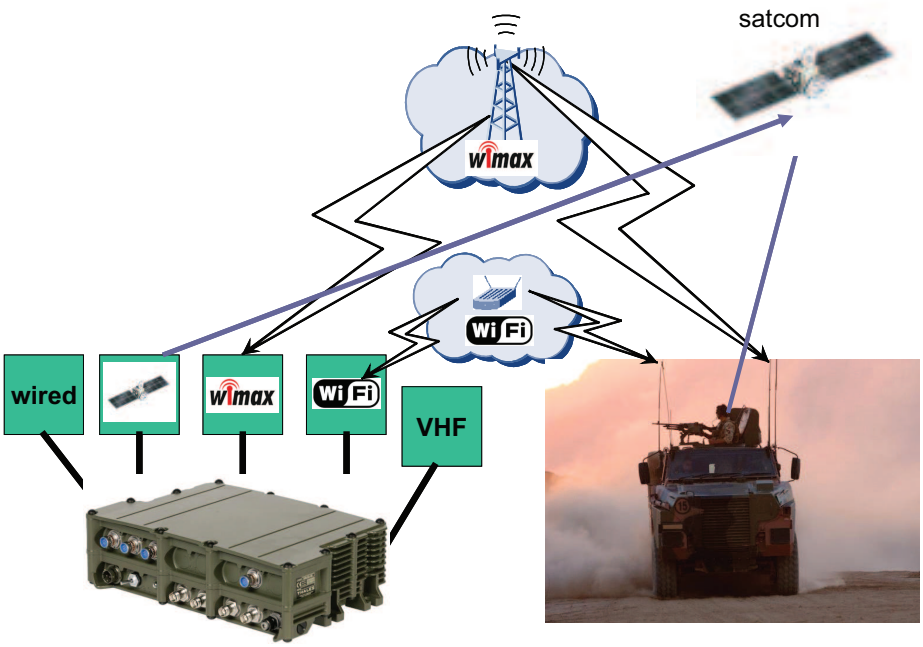


Figure 2: Military application of concurrent access.

In the military context concurrent access delivers an additional advantage:

- Security, when tactical information is transferred it is crucial that no hostile party can intercept this information. Using concurrent access the information can be split over multiple connections. With multiple connections over different wireless communication networks the hostile party has to intercept more communication networks.

1.4 How does Concurrent Access work?

Concurrent access does deliver a lot of advantages in terms of speed, reliability and security. If a device is equipped with multiple antennas, some intelligence has to be developed for enabling the advantages of concurrent access. Figure 3 provides a schematic overview of how concurrent access can be applied.

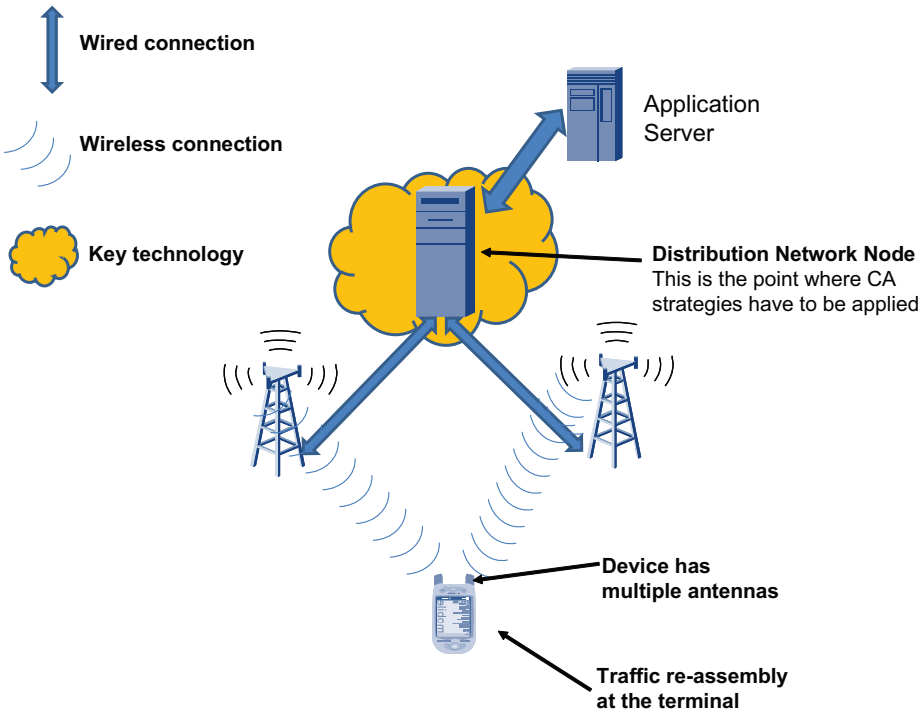


Figure 3: How concurrent access works.

The application server typically contains webpages (HTTP) or files (FTP) that will be requested by clients. Concurrent access enables performance improvements that will reduce the access time and file transfer time. In performance model terms the file transfer time is modelled by the sojourn time. This is the time between the moment of which a file is requested and the moment the file transfer is completed. The orange cloud contains the distribution node. The distribution node implements a concurrent access strategy that optimize the performance for all concurrent access clients. In Figure 1 the different wireless technologies are connected to the IP network. All the routers and operator antennas that connect the wireless technologies to the IP network will be denoted by a wireless node or node. In the concurrent access setting two types of clients have to be distinguished:

- Foreground traffic which consists of clients that have multiple antennas and use multiple wireless connections simultaneously,
- Background traffic which consists of clients that only use one wireless connection.

The aim of this thesis is to develop strategies that will optimize the performance for foreground traffic. Concurrent access strategies can be divided in two classes:

- Server selection
- Job split

A **server selection** strategy will select for each requested file a wireless node for transfer. The node will not be changed during transfer. Figure 4 provides a schematic overview for the application of server selection for two wireless nodes:

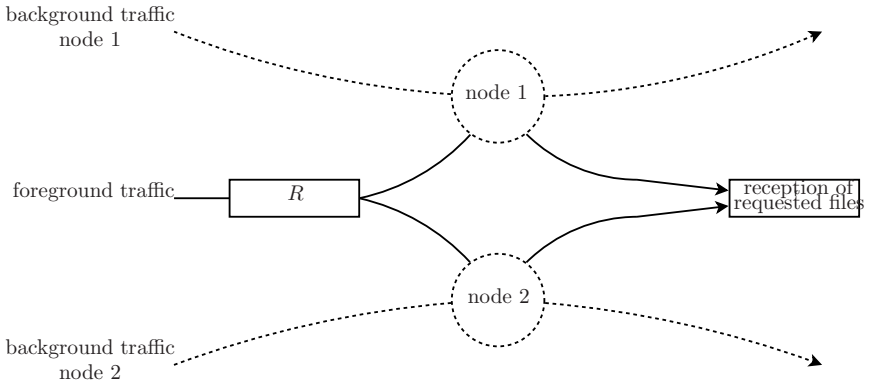


Figure 4: Static server selection. R denotes the server selection strategy that is used.

Server selection can be applied statically or dynamically. With a static server selection strategy R for each foreground file request a node is selected for the file transfer independent of the state of the system. Examples of static server selection policies are

- Round Robin (RR),
- Random server selection.

With Round Robin the server selection is applied using a fixed sequence, for example $1, 2, 1, 2$ (the numbers correspond to the selected node number). With random server selection upon arrival a node is selected stochastically. The selection probabilities p_i are determined by the selected probability distribution over the nodes.
Dynamic server selection is a server selection strategy based on dynamic observations. This can be the file size or the number of flows on the nodes. If there are two nodes with n_1 and n_2 the number of flows on server 1 and server 2, the strategy will be a function $R(n_1, n_2)$ with for each combination of n_1 and n_2 an optimal decision.
A **job split** strategy will split a requested file over multiple networks. For job split the optimal strategy comprises the optimal splitting of files. Figure 5 provides a schematic overview for concurrent access using job split for two nodes. In this figure the arrows are joined together because the splitted files have to be recombined at the terminal. In the two node case the splitting strategy can be identified using a splitting factor α. Ideally this should be done such that latency due to reassembly is avoided.

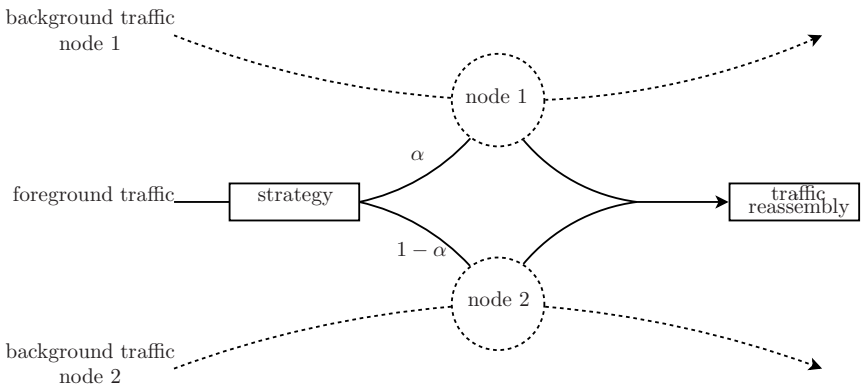


Figure 5: Job split.

1.5 Research questions

In the setting that is described in Section 1.4 several questions were formulated. The research questions are:

What is the impact of burstiness in arrivals on the optimal policies? Internet traffic does not exactly behave like a Poisson process. Internet traffic arrives in bursts. This is for example caused by correlation between files loaded together on a webpage. Furthermore during a day there are 'busy hours' where lots of traffic is generated for example people that go online during lunch time.

What is the impact of file size distributions on the optimal policies? In the most simple traffic models file sizes are modeled by an exponential distribution. For internet traffic file sizes are much 'more variable' than exponentially distributed files. Therefore it is important to find out the impact of filesize distribution on finding optimal policies.

What is the impact of partial observability? The optimization models are formulated on the assumption that only the total number of flows can be observed and not the type of flow. However for the optimization there is a difference between foreground flows which are the clients that will use CA and background flows which are clients that only use one wireless network.

What performance improvement gain can be obtained by taking in account the history of observations? With the partial observation problem the sequence of observations (on the number of flows for each node) provides more information than individual observations. Therefore the use of observation history can improve the performance of strategies that are based on partial observability.

How to develop simple heuristics for optimal splitting? Calculation of optimal policies is in general a time and memory consuming operation. Can this be simplified using a simple rule that is close to the optimal solution?

1.6 Approach

For the answer to the research questions the dynamic server selection problem will be modeled as Markov Desicion Problems (MDPs). These problems will be formulated for the different combinations of burstiness and different file size distributions. The MDPs will be solved using backwards recursion. The resulting policies will be implemented and validated in OPNET, a simulation environment that incorporates IP and network specific dynamics.

1.7 Structure of the thesis

This thesis addresses dynamic server selection strategies. In Chapter 2 dynamic server selection is studied using MDPs (Markov Decision Processes). These MDPs generate optimal decision strategies based on the number of flows on the wireless nodes for different file size distributions, including Erlang, exponential and hyper-exponential distributions. The state spaces for the models contain more dimensions than the number of flows on the nodes. Therefore a partial observation approach is proposed using a distribution conditioned on the observed number of flows. In Chapter 3 the partial observation problem of Chapter 2 is approached by applying Bayesian dynamic programming. In Chapter 4 an algorithm is proposed that can be used for calculating the expected sojourn time conditioned on the number of flows. Furthermore, a strategy is proposed that is based on known result on the conditional sojourn time for a single PS node. All the different decision models have been implemented in a realistic simulation environment. Chapter 5 presents the results in terms of performance for the different dynamic server selection strategies that are discussed in this thesis. The resulting strategies of Chapters 2, 3 and 4 were all evaluated together, therefore these results are combined together in Chapter 5. Figure 6 provides a graphical overview on how the chapters of this thesis are index related:

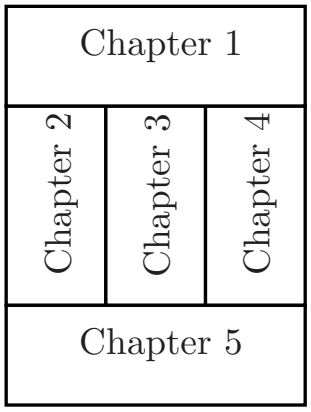


Figure 6: Relation between chapters.

2 Dynamic splitting of traffic with MMPP arrivals

In this chapter models are formulated that will generate policies for traffic including burstiness arrivals and different file size distributions. These models will be used for experiments on the impact of burstiness in arrivals and the impact of file size distributions on the optimal policies.
For the calculation of optimal policies MDP's will be formulated and solved using backward recursion. Before MDP's can be formulated some modelling assumptions have to be made. The assumptions and their motivation are described in Section 2.1. In Section 2.2 a basic (MDP) model is formulated which optimizes foreground traffic for Poisson arrivals and exponentially distributed file sizes. The basic MDP model is a template that will be extended with more complex assumptions including partial observation, different arrival processes and different file sizes. In Section 2.3 an approach for partial observation, based on conditioning over the possible observations, is presented. Section 2.4 will extend the basic MDP model in Section 2.2 applying the partial observation approach in Section 2.3. In Section 2.5 the partial observation model with Poisson arrival and exponential file sizes of Section 2.4 will be extended with Erlang and hyper-exponential file sizes. With the final extension in section 2.6 the partial observation models of Section 2.4 and Section 2.5 are extended with MMPP arrivals. For the MMPP models burstiness of traffic is parametrized in a interrupted Poisson process (IPP) with two parameters. Using the MDP models in Section 2.7 various optimal policies have been calculated for different file sizes and different burstiness parameters for the MMPP arrivals and the resulting optimal policies are presented and compared.

2.1 Model assumptions

For the optimization of concurrent access different models are defined. These models are based on the following assumptions:

Nodes Wireless network nodes can be modeled as PS (processor sharing) nodes. Modeling network performance using PS based models [2, 12, 7] is applicable to a variety of communication networks, including CDMA 1xEV-DO, WLAN, and UMTS-HSDPA. In fact, PS models can actually model file transfers over WLANs accurately [12], hence taking into account the complex dynamics of the file transfer application and its underlying protocol-stack, including their interactions.

Arrivals The arrivals of download requests are modeled as Poisson processes and Markov Modulated Poisson Processes. (See Section 2.6).

File sizes The file sizes are modeled in the service time distribution of the PS nodes.

Observability On each wireless network node only the total number of flows can be observed. No distinction can be made between the traffic types (foreground or background).

2.2 Basic model

Consider the situation where two connections are available and a file has to be transferred. Traffic consists of the transfer of files where each transfer in progress is denoted by a flow. For both nodes the amount of traffic can be observed (number of flows). The traffic on the nodes consists of clients that only use one node. This traffic is identified as background traffic. Besides background traffic there is foreground traffic from users that want to optimize the transfer speed by using two nodes. This type of traffic is identified as foreground traffic. The objective is to minimize the expected sojourn time of the foreground traffic files that have to be transmitted given the total amount of (foreground + background) flows on both connection-nodes. Figure 4 provides insight in the process that is being optimized. In this model, one has to decide, based on the total observed number of flows on each server, which node should be selected for the file transfer.

The server selection setting can be modeled as a Semi-Markov Decision Process (SDMP). In Appendix 8.A.1 more can be found about SDMP's. The model is visualized in Figure 7. Here background flow arrivals are modeled as Poisson processes with arrival rates λ_1 and λ_2 respectively. The arrival of foreground files is modeled as a Poisson process with arrival rate λ_0. The connection nodes can be modeled as PS (Processor Sharing) nodes with exponentially distributed service times (all users that use a connection have to equally share the bandwidth with the other users). In this case the service times correspond with the file size distribution. For now all streams have the same exponentially distributed file sizes with expected file size $\frac{1}{\mu}$ Mbit. Each node has a bandwidth or capacity indicated by C_i Mbps. Furthermore the system has statespace $\mathcal{S}$. $R(s)$ represents the dynamic policy based on system state $s \in \mathcal{S}$.

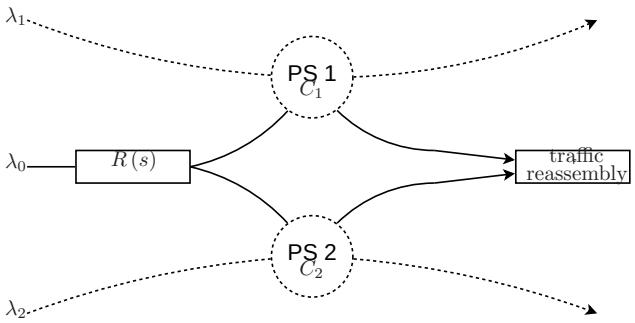


Figure 7: Server selection model with:

$s \in \mathcal{S}$ is the state of the system (# of files),

$R(s)$ is the dynamic policy based on state s,

C_i the capacity on node i,

$\frac{1}{\mu}$ the average file size,

λ_0 foreground arrival rate,

λ_i background arrival rate on node i.

When the amount of foreground traffic is relatively low compared to background traffic this can have negative impact on the performance of foreground traffic. Therefore a model should be formulated that takes into account the distinction between foreground traffic and background traffic. The state space of this model is $S = {\mathbb{N}_0}^4$ where in $s = (b_1, f_1, b_2, f_2) \in \mathcal{S}$. In this state space f_i denotes the number of foreground flows in connection node i and b_i denotes the number of foreground flows in connection node i. For each arriving foreground flow, a connection has be selected for transmission. This selection is modeled in the action space $a \in \mathcal{A} = \{1, 2\}$ with 1,2 the index of the connection node that should be selected. The idea is to minimize the expected sojourn time for foreground flows. The expected sojourn time can be derived from the expected number of flows using Little's formula:

$$\lambda \mathbb{E}[S] = \mathbb{E}[L] \tag{1}$$

with:

λ	the (foreground traffic) arrival rate,
$\mathbb{E}[S]$	the expected sojourn time and
$\mathbb{E}[L]$	the steady state expected number of foreground flows.

The expected number of flows can be optimized by choosing the reward $r(s, a) = f_1 + f_2$ equal to the number of foreground flows. Now the total expected reward g corresponds to the expected number of foreground flows in the system. As the average number of flows should be minimized the value iteration will become

a minimization problem. This description can be converted in a MDP which can be solved using value iteration as described in Appendix 8.A.2. Using the value iteration definition in Equation (72) from Appendix 8.A.2 the backward recursion equations can be defined with $s \in \mathcal{S}$, $t = 0, 1, \cdots$ and $\tau = \frac{1}{\gamma}$:

$$V_{(t+1)\tau}(s) = \frac{f_1 + f_2}{\gamma} \tag{2a}$$

$$+ \min_{a \in \{1,2\}} \left[\frac{\lambda_0}{\gamma} V_{t\tau}(s + e_{2a}) \right] \tag{2b}$$

$$+ \sum_{i \in \{1,2\}} \left[\frac{\lambda_i}{\gamma} V_{t\tau}(s + e_{2i-1}) \right] \tag{2c}$$

$$+ \sum_{i \in \{1,2\}} \left[d_{b,i} \frac{\mu_i}{\gamma} C_i V_{t\tau} \left([s - e_{2i-1}]^+ \right) \right] \tag{2d}$$

$$+ \sum_{i \in \{1,2\}} \left[d_{f,i} \frac{\mu_0}{\gamma} C_i V_{t\tau} \left([s - e_{2i}]^+ \right) \right] \tag{2e}$$

$$+ \sum_{i \in \{1,2\}} \left[(1 - d_{b,i}) \frac{\mu_i}{\gamma} C_i + (1 - d_{f,i}) \frac{\mu_0}{\gamma} C_i \right] V_{t\tau}(s), \tag{2f}$$

with

$$d_{b,i} = \begin{cases} \dfrac{b_i}{b_i + f_i} & \text{if} \quad b_i + f_i > 0 \\ 0 & \text{otherwise} \end{cases},$$

$$d_{f,i} = \begin{cases} \dfrac{f_i}{b_i + f_i} & \text{if} \quad b_i + f_i > 0 \\ 0 & \text{otherwise} \end{cases},$$

and

$$\gamma = \lambda_0 + \sum_{i \in \{1,2\}} \left[\lambda_i + (\mu_i + \mu_0) C_i \right].$$

In Equation (2):

(2a) corresponds to the number of foreground flows in state s,

(2b) corresponds to the event of a foreground flow arrival transition,

(2c) corresponds to the event of with a background flow arrival transition,

(2d) corresponds to the event of with a background flow completion,

(2e) corresponds to the event of with a foreground flow completion and

(2f) corresponds to the dummy transition.

Furthermore e_{2i-1} is the unit vector that has zeros on all dimensions except on the dimension that corresponds to the number of background flows on server

i. e_{2i-1} does the same for foreground traffic. On that position the vector has value 1. This vector is used in the equations for identifying the transitions of the Markov Chain. The fractions $\frac{b_i}{b_i+f_i}$ and $\frac{f_i}{b_i+f_i}$ are a result of the fact that the connection nodes are modeled as PS (processor sharing) nodes. Each transfer will receive a fraction $\frac{1}{b_i+f_i}$ of service (the number of flows in a connection node is the sum of background and foreground type traffic). From this the total fraction of service to transfer on node i will be $\frac{b_i}{b_i+f_i}$ for background traffic and $\frac{f_i}{b_i+f_i}$ for foreground traffic. As the total expected reward g corresponds to the expected total number of foreground flows Little's formula can now be applied in order to obtain the expected sojourn time:

$$\mathbb{E}[S] = \frac{\mathbb{E}[L_f]}{\lambda_0} = \frac{g}{\lambda_0}. \tag{3}$$

In Equation (3):

$\mathbb{E}\{S\}$	is the expected foreground flow sojourn time,
$\mathbb{E}\{L_f\}$	is the expected number of foreground flows,
λ_0	is the foreground flow arrival intensity and
g	is the long-term average expected reward (in this case expected number of foreground flows).

In this model the decisions are based on knowledge on both the number of foreground and background transfers in the two connection nodes. This is however not always the case. Therefore a partial observation approach has to be used.

2.3 Partial observation

In practice only the total number of flows on the connection nodes (i.e., $b_i + f_i$ for node i, $i = 1, 2$) can be observed. Furthermore the MDP models for more complex file sizes do contain more dimensions than can be observed. This introduces the problem of partial observation. The problem of partial observation is introduced on the assumption that only the total amount of traffic can be observed and decisions can only be based on observations. For the clear definition of the partial observation in the models we define an observation space $O = {\mathbb{N}_0}^2$ with $(o_1, o_2) \in O$ with o_i the observed total number of flows on connection node i. The observed states are compound states, the observation is the sum of the number of flows in the non observable states. In the case of the basic model, with Poisson arrivals and exponential file sizes, when the distinction between background and foreground traffic is made in the MDP model the state space will become $S = {\mathbb{N}_0}^4$ where $(b_1, f_1, b_2, f_2) = s \in \mathcal{S}$. In the state space f_i corresponds to the number of foreground traffic flows on connection-node with index $i \in \{1, 2\}$ and b_i corresponds to the number of background flows on node i. With this higher dimensional model multiple combinations of states $s \in \mathcal{S}$ can correspond to an given observation $o \in O$. For example: with the foreground optimizing model for the observation $(o_1, o_2) = (2, 2)$ the system can be in any of the following states:

$$\left\{(b_1,f_1,b_2,f_2)\,|_{b_2+f_2=2}^{b_1+f_1=2}\right\} = \left\{\begin{array}{c}(0,2,0,2),(1,1,0,2),(2,0,0,2),\\(0,2,1,1),(1,1,1,1),(2,0,1,1),\\(0,2,2,0),(1,1,2,0),(2,0,2,0)\end{array}\right\}. \quad (4)$$

Now nine states will all be observed as $(2,2)$. As a consequence, the decision should be based on all the possible states that correspond to this observation. Consider the MDP model in Equation (2). For the partially observable model $R(o)$ is the policy, the set of actions given the system is observed in $o \in O$. $R_{t\tau}(o)$ is defined as the policy at time $t\tau$ in the value iteration algorithm. For a given policy $R_{t\tau}(o)$ at epoch $t\tau$ the Markov chain given this policy can be solved. $R_{t\tau}(o)$ is based on the observation o however because o is composed of possible states $s \in \mathcal{S}$, o can be written as a function of s:

$$o(s) = \{o|q'(o,s) = 1, s \in \mathcal{S}, o \in O\}, \quad (5)$$

where $q'(o,s)$ is the probability of observing o in s:

$$q'(o,s) = \mathbb{P}(o|s, o \in O, s \in \mathcal{S}). \quad (6)$$

For the model in this paragraph $o(s)$ is defined by:

$$o(b_1,f_1,b_2,f_2) = (b_1+f_1, b_2+f_2). \quad (7)$$

With this function $R_{t\tau}(o)$ can be written as a policy on s, $R_{t\tau}(o|s) = R_{t\tau}(o(s))$. Denote $\nu_{R_{t\tau}}$ as the time limiting distribution on statespace $\mathcal{S}$ for the policy $R_{t\tau}$ on epoch $t\tau$ of the value iteration algorithm. So $\nu_{R_{t\tau}}(s)$ is the probability distribution over the complete state space in the value iteration algortihm at epoch $t\tau$. The distribution $\nu_{R_{t\tau}}$ at epoch $t\tau$ can be found by plugging in the policy at $t\tau$, $R_{t\tau}$ in the Markov chain. The solution of the Markov chain can be found using a numerical approch like SOR. When the distribution $\nu_{R_{t\tau}}$ is found this distribution can be conditioned on the observation space $o \in O$, in which the policy is defined.

$$\nu_{R_{t\tau}}(s|o) = \left\{\begin{array}{lll}\frac{\nu_{R_{t\tau}}(s)}{\sum_s \nu_{R_{t\tau}}(s) q'(o,s)} & \text{if} & \sum_s \nu_{R_{t\tau}}(s) q'(o,s) > 0\\ 0 & \text{otherwise.} & \end{array}\right. \quad (8)$$

Using the policy conditioned on the observation space O a modified MDP can be solved:

$$\begin{aligned} R_{(t+1)\tau}(o) &= \underset{a\in\mathcal{A}}{\operatorname{argmax}}\left\{\sum_s \nu_{R_{t\tau}}(s|o)\cdot \right. \quad (9)\\ &\qquad \left.\left[r(s,a)\tau + q(s,a)\sum_{s'} p(s,a,s') V_{t\tau}(s') + (1-q(s,a)) V_{t\tau}(s)\right]\right\},\\ a(s) &= R_{t\tau}(o(s)),\\ V_{(t+1)\tau}(s) &= r(s,a(s))\tau + q(s,a(s))\sum_{s'} p(s,a,s') V_{t\tau}(s') + [1-q(s,a(s))] V_{t\tau}(s). \end{aligned}$$

Next sections will describe how the partial information approach based on the conditional distribution $\nu_{R_{t\tau}}(s|o)$ for three different file size distributions.

2.4 Poisson arrivals with exponential file sizes

As described in the start of this chapter, only the number of flows on each connection node can be observed. The distribution of background and foreground transfers cannot be observed. However the framework presented in Section 2.3 can be applied to the fully observable MDP. As described earlier there is an observation space $O = {\mathbb{N}_0}^2$ with $(o_1, o_2) \in O$ with o_i the observed total number of flows on connection node i. Furthermore the state space will become $S = {\mathbb{N}_0}^4$ where $(b_1, f_1, b_2, f_2) = s \in \mathcal{S}$. In the state space f_i corresponds to the number of foreground traffic flows on connection-node $i \in \{1, 2\}$ and b_i corresponds to the number of background flows on node i. e_{2i-1} is the unit vector that has zeros on all dimensions except on the dimension that corresponds to the number of backround flows on server i. e_{2i} does the same for foreground traffic. The cost function is defined by

$$r(s) = f_1 + f_2. \tag{10}$$

The value iteration algorithm will produce a policy R_o which contains all decisions that should be made when o flows are observed on the nodes and is defined by:

$$\begin{aligned}
R_{t\tau}(o) &= \underset{a\in\{1,2\}}{\operatorname{argmin}} \left\{ \sum_s \nu_{R_{t\tau}}(s|o)\, V_{t\tau}(s + e_{2a}) \right\}, \\
a(s) &= R_{t\tau}(o(s)), \\
V_{(t+1)\tau}(s) &= \frac{f_1 + f_2}{\gamma} && \text{(11a)} \\
&\quad + \frac{\lambda_0}{\gamma} V_{t\tau}\left(s + e_{2a(s)}\right) && \text{(11b)} \\
&\quad + \sum_{i\in\{1,2\}} \left[\frac{\lambda_i}{\gamma} V_{t\tau}(s + e_{2i-1}) \right] && \text{(11c)} \\
&\quad + \sum_{i\in\{1,2\}} \left[d_{b,i} \frac{\mu_i}{\gamma} C_i V_{t\tau}\left([s - e_{2i-1}]^+ \right) \right] && \text{(11d)} \\
&\quad + \sum_{i\in\{1,2\}} \left[d_{f,i} \frac{\mu_0}{\gamma} C_i V_{t\tau}\left([s - e_{2i}]^+ \right) \right] && \text{(11e)} \\
&\quad + \sum_{i\in\{1,2\}} \left[(1 - d_{b,i}) \frac{\mu_i}{\gamma} C_i + (1 - d_{f,i}) \frac{\mu_0}{\gamma} C_i \right] V_{t\tau}(s), && \text{(11f)} \\
& && \text{(11g)}
\end{aligned}$$

with

$$d_{b,i} = \begin{cases} \dfrac{b_i}{b_i + f_i} & \text{if} \quad b_i + f_i > 0 \\ 0 & \text{otherwise} \end{cases},$$

$$d_{f,i} = \begin{cases} \dfrac{f_i}{b_i + f_i} & \text{if} \quad b_i + f_i > 0 \\ 0 & \text{otherwise} \end{cases},$$

and

$$\gamma = \lambda_0 + \sum_{i \in \{1,2\}} \left[\lambda_i + (\mu_i + \mu_0) C_i\right],$$

with parameters:

λ_0 the arrival rate for foreground flows,

λ_1 the arrival rate for background traffic on node 1,

λ_2 the arrival rate for background traffic on node 2,

μ_0 file-size distribution parameter for foreground flows,

μ_1 file-size distribution parameter for background flows on node 1,

μ_2 file-size distribution parameter for background flows on node 2,

C_1 the capacity for node 1,

C_2 the capacity for node 2.

The function o maps the number of flows (b_1, f_1, b_2, f_2) to the corresponding observation (o_1, o_2) of the total number of flows:

$$o\left(b_1, f_1, b_2, f_2\right) = \left(b_1 + f_1, b_2 + f_2\right).$$

In Equation (11):

(11a) corresponds to the number of foreground flows in state s,

(11b) corresponds to the event of a foreground flow arrival,

(11c) corresponds to the event of a background flow arrival,

(11d) corresponds to the event of a background flow completion,

(11e) corresponds to the event of a foreground flow completion and

(11f) corresponds to the dummy transition.

Note that for the server selection models, the direct reward $r\left(s, a\right)$ and transition rate for the arrivals of foreground traffic λ_0 do not depend on choice a. Therefore the optimal decision can be directly based on the value function of the state that will be reached with decision a.

Usually with value iteration the total average expected reward g can be derived by the limit $\frac{V_{(t+1)\tau}(s) - V_{t\tau}(s)}{\tau} \to g$. The calculation of $\nu_{R_{t\tau}}$ enables the direct calculation of g from the policy:

$$g = \sum_{s \in \mathcal{S}} \nu_{R_{t\tau}}(s)\, r\left(s, R_{t\tau}\left(o\left(s\right)\right)\right). \tag{12}$$

2.5 Model extension 1: different file size distributions

The basic model including partial observation with Poisson arrivals and exponential file sizes is defined for now. The exponential distribution has a squared coefficient of variation that is equal to 1. One of the research questions is about finding out the impact of the file size distribution. Therefore the model will be extended with hyper-exponential and Erlang file sizes.

2.5.1 Poisson arrivals with hyper-exponential file sizes

The model with hyper-exponential file sizes is using the hyper-exponential distribution, with balanced means. A hyper-exponential distribution H_k consists of k exponential distributions with different rates μ_i. Each exponential distribution i has a selection probability $p_i > 0$ with $\sum_{i=1}^{k} p_i = 1$. In other words the exponential distributions in the hyper-exponential distributions can be considered as multiple 'classes' with their corresponding selection probabilities p_i. Each exponential distribution will contribute to the expectation of the hyper-exponential distribution with $\frac{p_i}{\mu_i}$, $\mathbb{E}[X] = \sum_{i=1}^{k} \frac{p_i}{\mu_i}$ ([10], 446). For the models the H_2-distribution is used with balanced means. A hyper-exponential distribution has balanced means when all the 'class' contributions to the expectation are equalized, $\frac{p1}{\mu_1} = \frac{p2}{\mu_2} = \cdots = \frac{pk}{\mu_k} = \frac{1}{k\mu}$. When the expectation and the squared coefficient of variation are provided for the H_2-distribution the selection probabilities p_i and the rates μ_i are fixed ([3], 13):

$$\begin{aligned} p_1 &= \frac{1}{2} - \frac{1}{2}\sqrt{\frac{c^2-1}{c^2+1}}, \\ p_2 &= \frac{1}{2} + \frac{1}{2}\sqrt{\frac{c^2+1}{c^2+1}}, \\ \mu_1 &= p_1 2\mu, \\ \mu_2 &= p_2 2\mu. \end{aligned} \tag{13}$$

The hyper-exponential distribution is only suitable for squared coefficient of variation $c^2 \geq 1$. Because the hyper-exponential distribution consists of multiple independent exponential distributions a multi-dimensional Markov Chain can be created. The next model is an extension of the exponential MDP where the extra dimensions for the H_2-distribution are added.

Define state space $S = {\mathbb{N}_0}^8$ with

$$(b_{1,1}, f_{1,1}, b_{1,2}, f_{1,2}, b_{2,1}, f_{2,1}, b_{2,2}, f_{2,2}) \in \mathcal{S}.$$

For the dimensions $b_{i,j}$ and $f_{i,j}$ the index $i \in \{1,2\}$ is the connection node index and $j \in \{1,2\}$ the index for the H_2 'classes'. $e_{4i+2j-5}$ is the unit vector that has zeros on all dimensions except on the dimension that corresponds to the number of foreground flows on server i and 'class' j. $e_{4i+2j-4}$ does the same for foreground traffic.
The value iteration equations are defined by:

$$
\begin{aligned}
R_{t\tau}(o) &= \underset{a\in\{1,2\}}{\operatorname{argmin}} \left\{ \sum_s \nu_{R_{t\tau}}(s|o) \sum_{j\in\{1,2\}} p_{0,j} V_{t\tau}\left(s + e_{4a+2j-4}\right), \right\} \\
a(s) &= R_{t\tau}\left(o(s)\right), \\
V_{(t+1)\tau}(s) &= \frac{f_{1,1}+f_{1,2}+f_{2,1}+f_{2,2}}{\gamma} && \text{(14a)} \\
&+ \sum_{j\in\{1,2\}} \left[\frac{\lambda_0}{\gamma} p_{0,j} V_{t\tau}\left(x + e_{4a(s)+2j-4}\right) \right] && \text{(14b)} \\
&+ \sum_{j\in\{1,2\}} \sum_{i\in\{1,2\}} \left[\frac{\lambda_i}{\gamma} p_{i,j} V_{t\tau}\left(s + e_{4i+2j-5}\right) \right] && \text{(14c)} \\
&+ \sum_{i,j\in\{1,2\}} \left[d_{b,i,j} \frac{\mu_{i,j}}{\gamma} C_i V_{t\tau}\left([s - e_{4i+2j-5}]^+\right) \right] && \text{(14d)} \\
&+ \sum_{i,j\in\{1,2\}} \left[d_{b,i,j} \frac{\mu_{0,j}}{\gamma} C_i V_{t\tau}\left([s - e_{4i+2j-4}]^+\right) \right] && \text{(14e)} \\
&+ \sum_{i,j\in\{1,2\}} \left[\left(1 - d_{b,i,j}\right) \frac{\mu_{i,j}}{\gamma} C_i + \left(1 - d_{f,i,j}\right) \frac{\mu_{0,j}}{\gamma} C_i \right] V_{t\tau}(s), && \text{(14f)}
\end{aligned}
$$

with

$$
d_{b,i,j} = \begin{cases} \dfrac{b_{i,j}}{b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2}} & \text{if} \quad b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2} > 0 \\ 0 & \text{otherwise} \end{cases},
$$

$$
d_{f,i,j} = \begin{cases} \dfrac{f_{i,j}}{b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2}} & \text{if} \quad b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2} > 0 \\ 0 & \text{otherwise} \end{cases},
$$

and

$$
\gamma = \sum_{i\in\{0,1,2\}} \left[\lambda_i\right] + \sum_{i,j\in\{1,2\}} \left[\left(\mu_{0,j} + \mu_{i,j}\right) C_i\right],
$$

with parameters:

λ_0 the arrival rate for foreground traffic,

λ_1 the arrival rate for background traffic on node 1,

λ_2 the arrival rate for background traffic on node 2,

${c_0}^2$ the squared coefficient of variation for foreground traffic,

${c_1}^2$ the squared coefficient of variation for background traffic on node 1,

${c_2}^2$ the squared coefficient of variation for background traffic on node 2,

$\mu_{0,j}$ the rates for foreground traffic with H_2 type j

$\mu_{i,j}$ the rates for background traffic on node i with H_2 type j

$p_{0,j}$ the probability of an 'class j' arrival for foreground traffic,

$p_{1,j}$ the probability of an 'class j' arrival for background traffic on node 1,

$p_{2,j}$ the probability of an 'class j' arrival for background traffic on node 2.

For this model the mapping from the state space to observation space is described by:

$$o\left(b_{1,1}, f_{1,1}, b_{1,2}, f_{1,2}, b_{2,1}, f_{2,1}, b_{2,2}, f_{2,2}\right) = \left(b_{1,1} + f_{1,1} + b_{1,2} + f_{2,2}, b_{2,1} + f_{2,1} + b_{2,2} + f_{2,2}\right).$$

In Equation (14):

(14a) corresponds to the number of foreground flows in state s,

(14b) corresponds to the event of a foreground flow arrival,

(14c) corresponds to the event of a background flow arrival,

(14d) corresponds to the event of a background flow completion,

(14e) corresponds to the event of a foreground flow completion and

(14f) corresponds to the dummy transition.

In this MDP formulation it can be found that the capacity on the PS nodes is equally divided over the hyper-exponential 'classes' j on node i with $\frac{b_{i,j}}{b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2}}$ for background traffic and $\frac{f_{i,j}}{b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2}}$ for foreground traffic.

2.5.2 Poisson arrivals with Erlang file sizes

As mentioned in Section 2.5.1 the hyper-exponential distribution is only suitable for modeling file sizes $c^2 \geq 1$. For file sizes $c^2 < 1$ the Erlang-distribution can be used. The Erlang-distribution with k phases E_k is the distribution of the sum of k independent identical exponential distributions. Each exponential distribution corresponds to a phase that has to be completed. Given the expectation in terms of processing/service rate $\frac{1}{\mu}$ the expectation of each phase is equal to $\frac{1}{k\mu}$. All phases have an exponential distribution with rate $k\mu$ ([10], 442). For the experiments an Erlang 2 file size distribution is used for $c^2 = \frac{1}{2} < 1$. The Erlang-2 distribution again causes the state space dimensions to double. Each dimension will be multiplied by two for the two phases of the Erlang-2 distribution. The phases are independent exponentially distributed which can be fitted again into a Markov Chain. For optimization with Erlang-2 file sizes a MDP is formulated with state space $S = {\mathbb{N}_0}^8$ where

$$\left(b_{1,1}, f_{1,1}, b_{1,2}, f_{1,2}, b_{2,1}, f_{2,1}, b_{2,2}, f_{2,2}\right) \in \mathcal{S}.$$

For the dimensions $b_{i,j}$ and $f_{i,j}$ the index $i \in \{1,2\}$ denotes the connection node index and $j \in \{1,2\}$ denotes E_2 phase index. $e_{4i+2j-5}$ and $e_{4i+2j-4}$ are unitvectors that identify the dimensions for the number of flows on the ith node in Erlang phase j. The value iteration equations are defined by:

$$
\begin{aligned}
R_{t\tau}(o) &= \underset{a\in\{1,2\}}{\operatorname{argmin}} \left\{ \sum_s \nu_{R_{t\tau}}(s|o)\, V_{t\tau}(s + e_{4a-2}) \right\}, \\
a(s) &= R_{t\tau}(o(s)), \\
V_{(t+1)\tau}(s) &= \frac{f_{1,1} + f_{1,2} + f_{2,1} + f_{2,2}}{\gamma} && \text{(15a)} \\
&\quad + \frac{\lambda_0}{\gamma} V_{t\tau}\left(s + e_{4a(s)-2}\right) && \text{(15b)} \\
&\quad + \sum_{i\in\{1,2\}} \frac{\lambda_i}{\gamma} V_{t\tau}(s + e_{4i-3}) && \text{(15c)} \\
&\quad + \sum_{i\in\{1,2\}} \left[d_{b,i,1} \frac{2\mu_i}{\gamma} C_i V_{t\tau}\left([s - e, b_{i,1} + e_{4i-1}]^+ \right) \right] && \text{(15d)} \\
&\quad + \sum_{i\in\{1,2\}} \left[d_{f,i,1} \frac{2\mu_0}{\gamma} C_i V_{t\tau}\left([s - e_{4i-2} + e_{4i}]^+ \right) \right] && \text{(15e)} \\
&\quad + \sum_{i\in\{1,2\}} \left[d_{b,i,2} \frac{2\mu_i}{\gamma} C_i V_{t\tau}\left([s - e_{4i-1}]^+ \right) \right] && \text{(15f)} \\
&\quad + \sum_{i\in\{1,2\}} \left[d_{f,i,2} \frac{2\mu_0}{\gamma} C_i V_{t\tau}\left([s - e_{4i}]^+ \right) \right] && \text{(15g)} \\
&\quad + \sum_{i,j\in\{1,2\}} \left[\left((1 - d_{b,i,j}) \frac{2\mu_i}{\gamma} + (1 - d_{f,i,1}) \frac{2\mu_0}{\gamma} \right) C_i \right] V_{t\tau}(s), && \text{(15h)}
\end{aligned}
$$

with

$$
d_{b,i,j} = \begin{cases} \dfrac{b_{i,j}}{b_{i,1} + f_{i,1} + b_{i,2} + f_{i,2}} & \text{if} \quad b_{i,1} + f_{i,1} + b_{i,2} + f_{i,2} > 0 \\ 0 & \text{otherwise} \end{cases},
$$

$$
d_{f,i,j} = \begin{cases} \dfrac{f_{i,j}}{b_{i,1} + f_{i,1} + b_{i,2} + f_{i,2}} & \text{if} \quad b_{i,1} + f_{i,1} + b_{i,2} + f_{i,2} > 0 \\ 0 & \text{otherwise} \end{cases},
$$

and

$$
\gamma = \sum_{i\in\{0,1,2\}} [\lambda_i] + \sum_{i,j\in\{1,2\}} [(\mu_i + \mu_0) C_i],
$$

with parameters:

λ_0 the arrival rate for foreground traffic,

λ_1 the arrival rate for background traffic on node 1,

λ_2 the arrival rate for background traffic on node 2,

$2\mu_0$ the rate for each Erlang phase for foreground traffic distribution i,

$2\mu_1$ the rate for each Erlang phase for background traffic distribution on node 1,

$2\mu_2$ the rate for each Erlang phase for background traffic distribution on node 2,

C_1 the capacity of node 1,

C_2 the capacity of node 2.

The mapping from state space to observation space is defined by:

$$o\left(b_{1,1}, f_{1,1}, b_{1,2}, f_{1,2}, b_{2,1}, f_{2,1}, b_{2,2}, f_{2,2}\right) = \left(b_{1,1} + f_{1,1} + b_{1,2} + f_{2,2}, b_{2,1} + f_{2,1} + b_{2,2} + f_{2,2}\right).$$

In Equation (15):

(15a) corresponds to the number of foreground flows in state s,

(15b) corresponds to the event of a foreground flow arrival,

(15c) corresponds to the event of a background flow arrival,

(15d) corresponds to the event of transition from Erlang-2 phase '1' to phase '2' for a background flow,

(15e) corresponds to the event of transition from Erlang-2 phase '1' to phase '2' for a foreground flow,

(15f) corresponds to the event of a background flow completion,

(15g) corresponds to the event of a goreground flow completion and

(15h) corresponds to the dummy transition.

The Erlang phase rates $2\mu_i$ are derived from the original file size parameters where the expected file size equals $\frac{1}{\mu_i}$ for $i \in \{0, 1, 2\}$. Within this MDP formulation $\frac{b_{i,j}}{b_{i,1}+b_{i,1}+b_{i,2}+b_{i,2}}$, $\frac{f_{i,j}}{b_{i,1}+b_{i,1}+b_{i,2}+b_{i,2}}$ is the equal partitioning of capacity on node i over the 2 phases j for foreground traffic and background traffic. Arrivals only occur to the first phase therefore the arrival part of the equation becomes

$$\frac{\lambda_0}{\gamma} V_{t\tau}\left(s + e_{4a(s)-2}\right) + \sum_{i \in \{1,2\}} \frac{\lambda_i}{\gamma} V_{t\tau}\left(s + e_{4i-3}\right).$$

In comparison to the (hyper)-exponential MDP formulations different transitions $V_{t\tau}\left(s - e, b_{i,1} + e_{4i-1}\right)$ and $V_{t\tau}\left(s - e_{4i-3} + e_{4i-1}\right)$ appear. This transitions correspond to the transition of phase 1 to phase 2 in the file size distributions.

2.6 Model extension 2: MMPP

In Sections 2.4 and 2.5 MDP models have been formulated under the assumption of different file size distributions. For modeling burstiness in traffic arrivals another model extension has to be made. Burstiness can be modeled using Markov Modulated Poisson Processes (MMPP).

Rationale for using MMPP When a file is requested / downloaded via HTTP or FTP usually a combination of files is downloaded. For example with a webpage request also additional images and other content is downloaded. With FTP when a directory is downloaded a number of files will be downloaded. This will create a burst of files that should be transferred and after the transfer a relatively long quiet period will occur. The arrival process can be modeled more realistically by using MMPP's. Furthermore busy hours (with higher traffic arrival intensity) can be modeled more realistically using MMPP. The advantage of MMPP is that more complex arrival processes can be caught while stil a Markov chain can be constructed. A disadvantage of MMPP is that the state space will grow fast when complex arrival processes are modeled.

MMPP definition A Markov Modulated Poisson Process (MMPP) is a multi state Poisson arrival process where the states correspond with different arrival rates. Consider a MMPP consisting m arrival rates $\lambda_1 \cdots \lambda_m$. In a MMPP the arrival rate state is determined by a m state continuous time Markov chain. The term Markov Modulated is the direct description of the Markov chain behind the arrival rate in a MMPP [11]. For the Markov chain that determines the arrival rate for a MMPP a generating matrix can be defined:

$$Q = \begin{bmatrix} -\omega_1 & \omega_{1,2} & \cdots & \omega_{1,m} \\ \omega_{2,1} & -\omega_2 & \cdots & \omega_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{m,1} & \omega_{m,2} & \cdots & -\omega_m \end{bmatrix}, \tag{16}$$

and

$$\omega_i = \sum_{j \neq i} \omega_{i,j}. \tag{17}$$

For a graphical example consider a three state MMPP with MMPP state $k \in \{1, 2, 3\}$ This Markov chain can be presented graphically as follows:

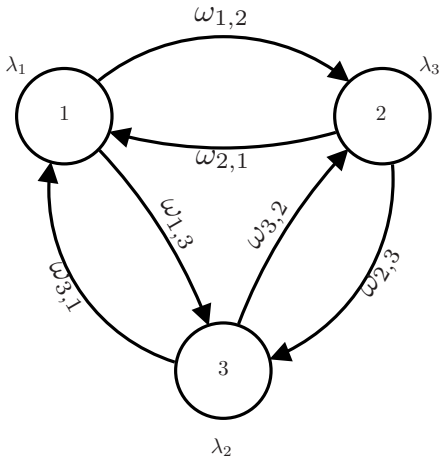


Figure 8: Example of a Markov chain for a Markov Modulated Poisson Process.

An interesting case of the example above is the Interrupted Poisson Process (IPP). In this case $m = 2$ resulting in the generating matrix:

$$Q = \begin{bmatrix} -\omega_1 & \omega_1 \\ \omega_2 & -\omega_2 \end{bmatrix}, \tag{18}$$

and corresponding arrival rate matrix

$$\Lambda = \begin{bmatrix} \lambda & 0 \\ 0 & 0 \end{bmatrix}. \tag{19}$$

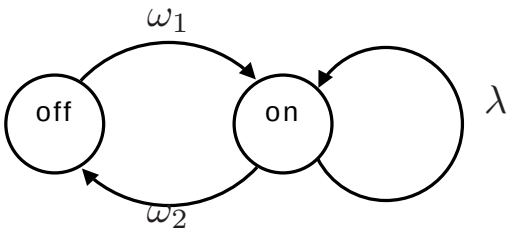


Figure 9: Interrupted Poisson Process.

In the MMPP cookbook [11] it is written that an IPP is stochastically equivalent to a Hyperexponential renewal process. Details can be found in Appendix 8.B.

2.6.1 Parametrization of burstiness

The models defined in Sections 2.4 and 2.5 will be extended with MMPP arrivals using a two state embedded Markov-chain for each arrival process. Each MMPP arrival process consists of a low arrival rate $\lambda_{i,L}$ or high arrival rate $\lambda_{i,L}$. Given

the two state MMPP transition matrix, $\begin{bmatrix} 0 & \omega_{i,L} \\ \omega_{i,H} & 0 \end{bmatrix}$, the average fraction of time that the arrival process is in a low or high state can be calculated:

$$\begin{aligned} p_{i,L} &= \frac{\omega_{i,H}}{\omega_{i,L} + \omega_{i,H}}, \\ p_{i,H} &= \frac{\omega_{i,L}}{\omega_{i,L} + \omega_{i,H}}. \end{aligned} \tag{20}$$

With these probabilities the average arrival intensity λ_i can be calculated using the corresponding arrival rates:

$$\begin{aligned} \lambda_i &= \lambda_{i,L} p_{i,L} + \lambda_{i,H} p_{i,H} \\ &= \lambda_{i,L} \frac{\omega_{i,L}}{\omega_{i,L} + \omega_{i,H}} + \lambda_{i,H} \frac{\omega_{i,H}}{\omega_{i,L} + \omega_{i,H}}. \end{aligned} \tag{21}$$

The rates of the two state MMPP transition matrix do not directly give an intuitive meaning to the specific MMPP. A better formulation of burstiness can be given in terms of time characteristics. Time characteristics off an MMPP arrival process can be formulated in two parameters:

- D duty cycle, the fraction of time the arrival process is in high state $D = p_{i,H}$. In other words, the concentratedness of the arrival bursts. The lower the duty cycle, the more concentrated the arrival bursts are.
- T cycle time, the average length if a high low cycle.

The rates for the MMPP embedded Markov chain can be obtained by transforming the parameters D and T in MMPP arrival rates

$$\begin{aligned} \lambda_{i,L} &= \frac{1}{(1-D)\,T}, \\ \lambda_{i,H} &= \frac{1}{(DT)}. \end{aligned} \tag{22}$$

With the time characteristics defined there is still some degree of freedom left in choosing the MMPP arrival rates $\lambda_{i,L}$ and $\lambda_{i,H}$. Multiple combinations of $\lambda_{i,L}$ and $\lambda_{i,H}$ lead to an equal average arrival rate:

$$\lambda_i = (1-D)\,\lambda_{i,L} + D\lambda_{i,H}. \tag{23}$$

In terms of $\lambda_{i,L}$ and $\lambda_{i,H}$ this will become

$$\begin{aligned} \lambda_{i,L} &= \frac{\lambda - D\lambda_{i,H}}{1-D}, \\ \lambda_{i,H} &= \frac{\lambda - (1-D)\,\lambda_{i,L}}{D}. \end{aligned} \tag{24}$$

When $\lambda_{i,L} = 0$ the MMPP will become an Interrupted Poisson Process and $\lambda_{i,H} = \dfrac{\lambda}{D}$. Now the burstiness can be defined in two parameters D and T.

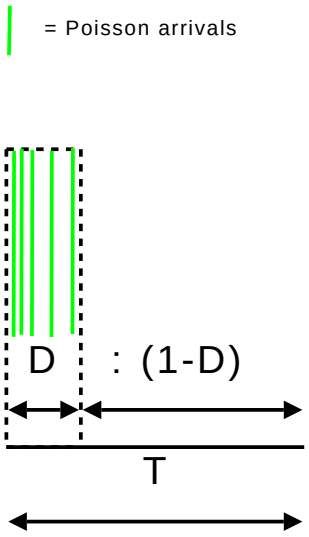
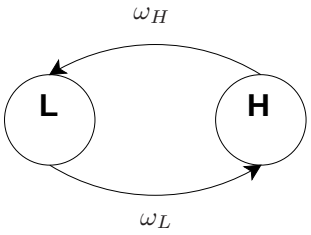


Figure 10: Duty cycle D fraction and cycle period T for IPP.

Figure 10 illustrates that the duty cycle actually corresponds to degree of concentration of the arrivals. In the interval T the expected number of arrivals remains equal. However when D becomes smaller the arrivals have to occur in a smaller time interval. As a consequence the arrival rate λ_H in the on state of the MMPP process is higher. When the duty cycle equals 100% the MMPP models are equal to the models with Poisson arrivals.

Transformation from Poisson to MMPP As described in Section 2.2 three arrival processes are modeled with rates λ_0, λ_1, λ_2 with λ_0 the arrival rate of foreground traffic and λ_1, λ_2 the arrival rates for the traffic on node 1 and 2. For the experiments on the impact of burstiness, the arrival processes will be extended into a two-state MMPP.



The MMPP model above has generating matrix and arrival matrix

$$Q = \begin{bmatrix} -\omega_L & \omega_L \\ \omega_H & -\omega_H \end{bmatrix}, \qquad \Lambda = \begin{bmatrix} \lambda_L & 0 \\ 0 & \lambda_H \end{bmatrix}. \tag{25}$$

Each of the three arrival processes $\lambda_0, \lambda_1, \lambda_2$ will be modeled by a two-state MMPP. Because the three arrival processes are chosen to be independent arrival processes three MMPPs have to be added to the model. The superposition

of the three MMPPs does result in an eight-state MMPP. The matrices for superposition of the three MMPP's can be archieved by applying the Kronecker sum $\oplus$. The Kronecker sum is defined by

$$A \oplus B = (A \otimes I_B) + (I_A \otimes B) \tag{26}$$

with I_A and I_B identity matrices that correspond to size to matrices A and B and $\otimes$ the Kronecker product.

$$C \otimes D = \begin{bmatrix} c_{1,1}D & c_{1,2}D & \cdots & c_{1,m}D \\ c_{2,1}D & c_{2,2}D & \cdots & c_{2,m}D \\ \vdots & \vdots & \ddots & \vdots \\ c_{n,1}D & c_{n,2}D & \cdots & c_{n,m}D \end{bmatrix}. \tag{27}$$

Given the generating matrices Q_0, Q_1, Q_2 and the corresponding rate matrices $\Lambda_0, \Lambda_1, \Lambda_2$ the resulting matrices become:

$$\begin{aligned} Q &= Q_0 \oplus Q_1 \oplus Q_2, \\ \Lambda &= \Lambda_0 \oplus \Lambda_1 \oplus \Lambda_2. \end{aligned} \tag{28}$$

This will become a $2 \times 2 \times 2 = 8$ dimensional state space where $k = 0, \cdots, 7$ identifies the MMPP state. Define $\omega_{i,j}$ the rates in the MMPP Markov chains for arrival process $i \in \{0, 1, 2\}$ and $j \in \{L, H\}$ the arrival process state. Further define $\lambda_{i,j}$ the corresponding arrival rates for process i in state j. Then the corresponding matrices will become:

$$Q_i = \begin{bmatrix} -\omega_{i,L} & \omega_{i,L} \\ \omega_{i,H} & -\omega_{i,H} \end{bmatrix}, \qquad \Lambda_i = \begin{bmatrix} \lambda_{i,L} & 0 \\ 0 & \lambda_{i,H} \end{bmatrix}. \tag{29}$$

The superposition of the individual generating matrices will produce the generating matrix:

$$Q = Q_0 \oplus Q_1 \oplus Q_2 \tag{30}$$
$$= \begin{bmatrix} -\omega_0 & \omega_{0,L} & \omega_{1,L} & 0 & \omega_{2,L} & 0 & 0 & 0 \\ \omega_{0,H} & -\omega_1 & 0 & \omega_{1,L} & 0 & \omega_{2,L} & 0 & 0 \\ \omega_{1,H} & 0 & -\omega_2 & \omega_{0,L} & 0 & 0 & \omega_{2,L} & 0 \\ 0 & \omega_{1,H} & \omega_{0,H} & -\omega_3 & 0 & 0 & 0 & \omega_{2,L} \\ \omega_{2,H} & 0 & 0 & 0 & -\omega_4 & \omega_{0,L} & \omega_{1,L} & 0 \\ 0 & \omega_{2,H} & 0 & 0 & \omega_{0,H} & -\omega_5 & 0 & \omega_{1,L} \\ 0 & 0 & \omega_{2,H} & 0 & \omega_{1,H} & 0 & -\omega_6 & \omega_{0,L} \\ 0 & 0 & 0 & \omega_{2,H} & 0 & \omega_{1,H} & \omega_{0,H} & -\omega_7 \end{bmatrix},$$

where

$$\begin{aligned} \omega_0 &= \omega_{0,L} + \omega_{1,L} + \omega_{2,L}, \\ \omega_1 &= \omega_{0,H} + \omega_{1,L} + \omega_{2,L}, \\ \omega_2 &= \omega_{0,L} + \omega_{1,H} + \omega_{2,L}, \\ \omega_3 &= \omega_{0,H} + \omega_{1,H} + \omega_{2,L}, \\ \omega_4 &= \omega_{0,L} + \omega_{1,L} + \omega_{2,H}, \\ \omega_5 &= \omega_{0,H} + \omega_{1,L} + \omega_{2,H}, \\ \omega_6 &= \omega_{0,L} + \omega_{1,H} + \omega_{2,H}, \\ \omega_7 &= \omega_{0,H} + \omega_{1,H} + \omega_{2,H}. \end{aligned}$$

Note that the transition rates have a nice structure. For the MMPP with arrival process $i = 0$ the 'jumps' in state space have size 1, for arrival process $i = 1$ size 2 and for arrival process $i = 2$ size 4. These jumps can be captured by $m_{i,k}$ where i is the arrival process for which the MMPP state changes and k is the current MMPP state:

Table 1: Transition pattern corresponding to MMPP state k.

k	$m_{0,k}$	$m_{1,k}$	$m_{2,k}$
0	+1	+2	+4
1	−1	+2	+4
2	+1	−2	+4
3	−1	−2	+4
4	+1	+2	−4
5	−1	+2	−4
6	+1	−2	−4
7	−1	−2	−4

The rates can not be superposed, because the arrival processes are three independent arrival processes. Therefore $\widehat{\Lambda}_i$ is introduced. $\widehat{\Lambda}_i$ corresponds to the arrival intensity of arrival process i when the superposed Markov chain of the three arrival processes is in state k.

$$
\begin{aligned}
\widehat{\Lambda}_0 &= \Lambda_0 \oplus 0_{Q_1} \oplus 0_{Q_2} \\
&= diag\left(\begin{bmatrix}\lambda_{0,L} & \lambda_{0,H} & \lambda_{0,L} & \lambda_{0,H} & \lambda_{0,L} & \lambda_{0,H} & \lambda_{0,L} & \lambda_{0,H}\end{bmatrix}\right), \\
\widehat{\Lambda}_1 &= 0_{Q_0} \oplus \Lambda_1 \oplus 0_{Q_2} \\
&= diag\left(\begin{bmatrix}\lambda_{1,L} & \lambda_{1,L} & \lambda_{1,H} & \lambda_{1,H} & \lambda_{1,L} & \lambda_{1,L} & \lambda_{1,H} & \lambda_{1,H}\end{bmatrix}\right), \\
\widehat{\Lambda}_2 &= 0_{Q_0} \oplus 0_{Q_1} \oplus \Lambda_2 \\
&= diag\left(\begin{bmatrix}\lambda_{2,L} & \lambda_{2,L} & \lambda_{2,L} & \lambda_{2,L} & \lambda_{2,H} & \lambda_{2,H} & \lambda_{2,H} & \lambda_{2,H}\end{bmatrix}\right).
\end{aligned} \tag{31}
$$

Here, 0_{Q_i} is the zero matrix with the dimensions of Q_i and all entries zero. When formulating MDP's with MMPP arrivals some functions should be formulated that identify the transition rates and arrival rates corresponding to MMPP state k:

Table 2: Rates corresponding to MMPP state k.

k	$\lambda_{0,k}$	$\lambda_{1,k}$	$\lambda_{2,k}$	$\omega_{0,k}$	$\omega_{1,k}$	$\omega_{2,k}$
0	$\lambda_{0,L}$	$\lambda_{1,L}$	$\lambda_{2,L}$	$\omega_{0,L}$	$\omega_{1,L}$	$\omega_{2,L}$
1	$\lambda_{0,H}$	$\lambda_{1,L}$	$\lambda_{2,L}$	$\omega_{0,H}$	$\omega_{1,L}$	$\omega_{2,L}$
2	$\lambda_{0,L}$	$\lambda_{1,H}$	$\lambda_{2,L}$	$\omega_{0,L}$	$\omega_{1,H}$	$\omega_{2,L}$
3	$\lambda_{0,H}$	$\lambda_{1,H}$	$\lambda_{2,L}$	$\omega_{0,H}$	$\omega_{1,H}$	$\omega_{2,L}$
4	$\lambda_{0,L}$	$\lambda_{1,L}$	$\lambda_{2,H}$	$\omega_{0,L}$	$\omega_{1,L}$	$\omega_{2,H}$
5	$\lambda_{0,H}$	$\lambda_{1,L}$	$\lambda_{2,H}$	$\omega_{0,H}$	$\omega_{1,L}$	$\omega_{2,H}$
6	$\lambda_{0,L}$	$\lambda_{1,H}$	$\lambda_{2,H}$	$\omega_{0,L}$	$\omega_{1,H}$	$\omega_{2,H}$
7	$\lambda_{0,H}$	$\lambda_{1,H}$	$\lambda_{2,H}$	$\omega_{0,H}$	$\omega_{1,H}$	$\omega_{2,H}$

$\omega_{i,k}$ is the transition rate out of MMPP state k for arrival process i,

$\lambda_{i,k}$ is the arrival rate for the arrival process i in MMPP state k.

2.6.2 MMPP arrivals with exponential file sizes

With the MMPP definition in Section 2.6.1 the models described in Section 2 can be extended with MMPP arrivals. The states for MMPP will add an extra dimension to the state space. Define $S \in \{0, \cdots, 7\} \times {\mathbb{N}_0}^4$ with $(k, b_1, f_1, b_2, f_2) \in \mathcal{S}$, where

k is the MMPP state as described in Section 2.6.1,

b_i is the number of background flows on node i,

f_i is the number of background flows on node i,

$m_{j,k}$ are the possible jumps for arrival process j (foreground and background arrivals) when the system is in MMPP state k,

$o(s)$ gives the observation that corresponds to Markov chain state $s \in \mathcal{S}$.

The value iteration equations with MMPP arrivals are defined by:

$$
\begin{aligned}
R_{t\tau}(o) &= \underset{a\in\{1,2\}}{\operatorname{argmin}} \left\{ \sum_s \nu_{R_{t\tau}}(s|o)\, V_{t\tau}(s+e_{2a}) \right\}, \\
a(s) &= R_{t\tau}(o(s)), \\
V_{(t+1)\tau}(s) &= \frac{f_1+f_2}{\gamma} && \text{(32a)} \\
&+ \frac{\lambda_{0,k}}{\gamma} V_{t\tau}\left(s+e_{2a(s)}\right) && \text{(32b)} \\
&+ \sum_{i\in\{1,2\}} \left[\frac{\lambda_{i,m}}{\gamma} V_{t\tau}(s+e_{2i-1}) \right] && \text{(32c)} \\
&+ \sum_{i\in\{1,2\}} \left[d_{b,i} \frac{\mu_i}{\gamma} C_i V_{t\tau}\left([s-e_{2i-1}]^+ \right) \right] && \text{(32d)} \\
&+ \sum_{i\in\{1,2\}} \left[d_{f,i} \frac{\mu_0}{\gamma} C_i V_{t\tau}\left([s-e_{2i}]^+ \right) \right] && \text{(32e)} \\
&+ \sum_{i\in\{0,1,2\}} \left[\frac{\omega_{i,k}}{\gamma} V_{t\tau}(s+m_{i,k}\cdot e_0) \right] && \text{(32f)} \\
&+ \sum_{i\in\{1,2\}} \left[(1-d_{b,i}) \frac{\mu_i}{\gamma} C_i + (1-d_{f,i}) \frac{\mu_0}{\gamma} C_i \right] V_{t\tau}(s) && \text{(32g)} \\
&+ \sum_{i\in\{0,1,2\}} \left[\frac{\omega_{i,L}+\omega_{i,H}-\omega_{i,k}+\lambda_{i,L}+\lambda_{i,H}-\lambda_{i,k}}{\gamma} \right] V_{t\tau}(s), && \text{(32h)}
\end{aligned}
$$

with

$$
d_{b,i} = \begin{cases} \dfrac{b_i}{b_i+f_i} & \text{if} \quad b_i+f_i>0 \\ 0 & \text{otherwise} \end{cases},
$$

$$
d_{f,i} = \begin{cases} \dfrac{f_i}{b_i+f_i} & \text{if} \quad b_i+f_i>0 \\ 0 & \text{otherwise} \end{cases},
$$

and

$$
\gamma = \sum_{i\in\{0,1,2\}} \left[\lambda_{i,L}+\lambda_{i,H}+\omega_{i,L}+\omega_{i,H} \right] + \sum_{i\in\{1,2\}} \left[(\mu_0+\mu_i) C_i \right],
$$

with parameters:

e_0 the unit vector with value 1 in the MMPP dimension 0 in the other dimensions,

e_{2i-1} the unit vector with value 1 in background dimension on node i and 0 in the other dimensions,

e_{2i} the unit vector with value 1 in foreground dimension on node i and 0 in the other dimensions,

$\lambda_{0,k}$ the arrival rate for foreground traffic when the system is MMPP state k,

$\lambda_{i,k}$ the arrival rate for background traffic when the system is MMPP state k,

$\omega_{0,k}$ the foreground traffic arrival intensity change rate when the system is MMPP state k,

$\omega_{i,k}$ the background traffic arrival intensity change rate when the system is MMPP state k,

μ_0 the file size parameter for foreground traffic,

μ_i the file size parameter for background traffic on node $i = 1, 2$,

C_i the capacity of node i.

The mapping from state space to observation space is defined by:

$$o\left(k, b_1, f_1, b_2, f_2\right) = \left(b_1 + f_1, b_2 + f_2\right).$$

In Equation (32):

(32a) corresponds to the number of foreground flows in state s,

(32b) corresponds to the event of a foreground flow arrival,

(32c) corresponds to the event of a background flow arrival,

(32d) corresponds to the event of a background flow completion,

(32e) corresponds to the event of a foreground flow completion,

(32f) corresponds to the event of a MMPP state transition (the transitions can be found in Table 1) and

(32g) corresponds to the dummy transitions.

Note that $m_{i,k} \cdot e_0$ corresponds to the possible jumps to other MMPP states when the system is in MMPP state k. The $\omega_{i,k}$ consists of values $\omega_{i,L}$ and $\omega_{i,H}$ according to the MMPP state k (see Table 2 in Section 2.6.1). The dummy transition $\omega_{i,L} + \omega_{i,H} - \omega_{i,k} + \lambda_{i,L} + \lambda_{i,H} - \lambda_{i,k}$ is added such that all rates will sum up to the uniformization parameter γ.

2.6.3 MMPP arrivals with hyper-exponential file sizes

For the hyper-exponential model define state space $S = \{0, \cdots, 7\} \times {\mathbb{N}_0}^8$ with dimensions $(k, b_{1,1}, f_{1,1}, b_{1,2}, f_{1,2}, b_{2,1}, f_{2,1}, b_{2,2}, f_{2,2} \in \mathcal{S})$ where

k is the MMPP state,

$b_{i,j}$ is the number of background flows on node i for 'H_2 class' j,

$f_{i,j}$ is the number of background flows on node i for 'H_2 class' j.

This model again uses balanced means H_2 as defined in equation (13) in Section 2.5.1. $i \in \{0,1,2\}$ is the index of the flow arrival type. The value iteration equations are defined by:

$$
\begin{aligned}
R_{t\tau}(o) &= \underset{a\in\{1,2\}}{\operatorname{argmin}} \left\{ \sum_{s} \nu_{R_{t\tau}}(s|o) \sum_{j\in\{1,2\}} p_{0,j} V_{t\tau}\left(s + e_{4a+2j-4}\right) \right\}, \\
a(s) &= R_{t\tau}\left(o(s)\right), \\
V_{(t+1)\tau}(s) &= \frac{f_{1,1}+f_{1,2}+f_{2,1}+f_{2,2}}{\gamma} && \text{(33a)} \\
&+ \sum_{j\in\{1,2\}} \left[\frac{\lambda_{0,k}}{\gamma} p_{0,j} V_{t\tau}\left(s + e_{4a(s)+2j-4}\right) \right] && \text{(33b)} \\
&+ \sum_{j\in\{1,2\}} \sum_{i\in\{1,2\}} \left[\frac{\lambda_{i,k}}{\gamma} p_{i,j} V_{t\tau}\left(s + e_{4i+2j-5}\right) \right] && \text{(33c)} \\
&+ \sum_{i,j\in\{1,2\}} \left[d_{b,i,j} \frac{\mu_{i,j}}{\gamma} C_i V_{t\tau}\left(\left[s - e_{4i+2j-5}\right]^{+}\right) \right] && \text{(33d)} \\
&+ \sum_{i,j\in\{1,2\}} \left[d_{f,i,j} \frac{\mu_{0,j}}{\gamma} C_i V_{t\tau}\left(\left[s - e_{4i+2j-4}\right]^{+}\right) \right] && \text{(33e)} \\
&+ \sum_{i\in\{0,1,2\}} \left[\frac{\omega_{i,k}}{\gamma} V_{t\tau}\left(s + m_{i,k}\cdot e_0\right) \right] && \text{(33f)} \\
&+ \sum_{i,j\in\{1,2\}} \left[\left(1 - d_{b,i,j}\right) \frac{\mu_{i,j}}{\gamma} C_i \right] V_{t\tau}(s) && \text{(33g)} \\
&+ \sum_{i,j\in\{1,2\}} \left[\left(1 - d_{f,i,j}\right) \frac{\mu_{0,j}}{\gamma} C_i \right] V_{t\tau}(s) && \text{(33h)} \\
&+ \sum_{i\in\{0,1,2\}} \left[\frac{\omega_{i,L}+\omega_{i,H}-\omega_{i,k}+\lambda_{i,L}+\lambda_{i,H}-\lambda_{i,k}}{\gamma} \right] V_{t\tau}(s), && \text{(33i)}
\end{aligned}
$$

with

$$
d_{b,i,j} = \begin{cases} \dfrac{b_{i,j}}{b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2}} & \text{if} \quad b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2} > 0 \\ 0 & \text{otherwise} \end{cases},
$$

$$
d_{f,i,j} = \begin{cases} \dfrac{f_{i,j}}{b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2}} & \text{if} \quad b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2} > 0 \\ 0 & \text{otherwise} \end{cases},
$$

and

$$
\gamma = \sum_{i\in\{0,1,2\}} \left[\lambda_{i,L}+\lambda_{i,H}+\omega_{i,L}+\omega_{i,H}\right] + \sum_{i,j\in\{1,2\}} \left[\left(\mu_{0,j}+\mu_{i,j}\right) C_i\right],
$$

with parameters:

e_0 the vector with value 1 in the MMPP dimension and 0 in the other dimensions,

$e_{4i+2j-5}$ the unit vector with value 1 in background dimension on node i, H_2 'class' j and 0 in the other dimensions,

$e_{4i+2j-4}$ the unit vector with value 1 in foreground dimension on node i, H_2 'class' j and 0 in the other dimensions,

$\lambda_{0,k}$ the arrival rate for foreground traffic when the system is MMPP state k,

$\lambda_{i,k}$ the arrival rate for background traffic when the system is MMPP state k,

$\omega_{0,k}$ the foreground traffic arrival intensity change rate when the system is MMPP state k,

$\omega_{i,k}$ the background traffic arrival intensity change rate when the system is MMPP state k,

$p_{0,j}$ the probability that a foreground arrival has 'H_2 class' j,

$p_{i,j}$ the probability that a background arrival on node i has 'H_2 class' j,

$\mu_{0,j}$ the file size parameter for foreground traffic for 'H_2 class' j,

$\mu_{i,j}$ the file size parameter for background traffic on node $i = 1, 2$ for 'H_2 class' j,

C_i the capacity of node i.

The mapping from state space to observation space is defined by:

$$o\left(k, b_{1,1}, f_{1,1}, b_{1,2}, f_{1,2}, b_{2,1}, f_{2,1}, b_{2,2}, f_{2,2}\right) = \left(b_{1,1} + f_{1,1} + b_{1,2} + f_{2,2}, b_{2,1} + f_{2,1} + b_{2,2} + f_{2,2}\right).$$

In equation (33):

(33a) corresponds to the number of foreground flows in state s,

(33b) corresponds to the event of a foreground flow arrival,

(33c) corresponds to the event of a background flow arrival,

(33d) corresponds to the event of a background flow completion,

(33e) corresponds to the event of a foreground flow completion,

(33f) corresponds to the event of a MMPP state transition (the transitions can be found in Table 1) and

(33g),(33h),(33i) correspond with the dummy transitions.

2.6.4 MMPP arrivals with Erlang file sizes

Define state space $S = \{0, \cdots, 7\} \times {\mathbb{N}_0}^8$ with

$$(k, b_{1,1}, f_{1,1}, b_{1,2}, f_{1,2}, b_{2,1}, f_{2,1}, b_{2,2}, f_{2,2}) \in \mathcal{S}.$$

The number of background flows is denoted by $b_{i,j}$ and the number of foreground flows is denoted by $f_{i,j}$. The index $i \in \{1, 2\}$ corresponds to the connection node index, $j \in \{1, 2\}$ corresponds to the index for the E_2 phases and $k \in \{0, \cdots, 7\}$ is the MMPP state. The value iteration equations with MMPP are defined by:

$$\begin{aligned}
R_{t\tau}(o) &= \underset{a\in\{1,2\}}{\operatorname{argmin}} \left\{ \sum_s \nu_{R_{t\tau}}(s|o)\, V_{t\tau}(s + e_{4a-2}) \right\}, \\
a(s) &= R_{t\tau}(o(s)), \\
V_{(t+1)\tau}(s) &= \frac{f_{1,1} + f_{1,2} + f_{2,1} + f_{2,2}}{\gamma} && \text{(34a)} \\
&+ \frac{\lambda_{0,k}}{\gamma} V_{t\tau}\left(s + e_{4a(s)-2}\right) && \text{(34b)} \\
&+ \sum_{i\in\{1,2\}} \left[\frac{\lambda_{i,k}}{\gamma} V_{t\tau}(s + e_{4i-3}) \right] && \text{(34c)} \\
&+ \sum_{i\in\{1,2\}} \left[d_{b,i,1} \frac{2\mu_i}{\gamma} C_i V_{t\tau}\left([s - e_{4i-3} + e_{4i-1}]^+ \right) \right] && \text{(34d)} \\
&+ \sum_{i\in\{1,2\}} \left[d_{f,i,1} \frac{2\mu_0}{\gamma} C_i V_{t\tau}\left([s - e_{4i-2} + e_{4i}]^+ \right) \right] && \text{(34e)} \\
&+ \sum_{i\in\{1,2\}} \left[d_{b,i,2} \frac{2\mu_i}{\gamma} C_i V_{t\tau}\left([s - e_{4i-1}]^+ \right) \right] && \text{(34f)} \\
&+ \sum_{i\in\{1,2\}} \left[d_{f,i,2} \frac{2\mu_0}{\gamma} C_i V_{t\tau}\left([s - e_{4i}]^+ \right) \right] && \text{(34g)} \\
&+ \sum_{i\in\{0,1,2\}} \left[\frac{\omega_{i,k}}{\gamma} V_{t\tau}(s + m_{i,k} \cdot e_k) \right] && \text{(34h)} \\
&+ \sum_{i,j\in\{1,2\}} \left[\left((1 - d_{b,i,j}) \frac{2\mu_i}{\gamma} + (1 - d_{f,i,j}) \frac{2\mu_0}{\gamma} \right) C_i \right] V_{t\tau}(s) && \text{(34i)} \\
&+ \sum_{i\in\{0,1,2\}} \left[\frac{\omega_{i,L} + \omega_{i,H} - \omega_{i,k} + \lambda_{i,L} + \lambda_{i,H} - \lambda_{i,k}}{\gamma} \right] V_{t\tau}(s), && \text{(34j)}
\end{aligned}$$

with

$$d_{b,i,j} = \begin{cases} \dfrac{b_{i,j}}{b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2}} & \text{if} \quad b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2} > 0 \\ 0 & \text{otherwise} \end{cases},$$

$$d_{f,i,j} = \begin{cases} \dfrac{f_{i,j}}{b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2}} & \text{if} \quad b_{i,1}+f_{i,1}+b_{i,2}+f_{i,2} > 0 \\ 0 & \text{otherwise} \end{cases},$$

and

$$\gamma = \sum_{i\in\{0,1,2\}} \left[\lambda_{i,L}+\lambda_{i,H}+\omega_{i,L}+\omega_{i,H}\right] + \sum_{i,j\in\{1,2\}} \left[\left(2\mu_0+2\mu_i\right)C_i\right],$$

with parameters:

e_0 the vector with value 1 in the MMPP dimension and 0 in the other dimensions,

$e_{4i+2j-5}$ the unit vector with value 1 in background dimension on node i, H_2 'class' j and 0 in the other dimensions,

$e_{4i+2j-4}$ the unit vector with value 1 in foreground dimension on node i, H_2 'class' j and 0 in the other dimensions,

$\lambda_{0,k}$ the arrival rate for foreground traffic when the system is MMPP state k,

$\lambda_{i,k}$ the arrival rate for background traffic when the system is MMPP state k,

$\omega_{0,k}$ the foreground traffic arrival intensity change rate when the system is MMPP state k,

$\omega_{i,k}$ the background traffic arrival intensity change rate when the system is MMPP state k,

μ_0 the file size parameter for foreground traffic,

μ_i the file size parameter for background traffic on node $i = 1, 2$,

C_i the capacity of node i.

The mapping from state space to observation space is defined by:

$$o\left(k, b_{1,1}, f_{1,1}, b_{1,2}, f_{1,2}, b_{2,1}, f_{2,1}, b_{2,2}, f_{2,2}\right) = \left(b_{1,1}+f_{1,1}+b_{1,2}+f_{2,2}, b_{2,1}+f_{2,1}+b_{2,2}+f_{2,2}\right).$$

In equation (34):

(34a) corresponds to the number of foreground flows in state s,

(34b) corresponds to the event of a foreground flow arrival,

(34c) corresponds to the event of a background flow arrival,

(34d) corresponds to the event of transition from Erlang-2 phase '1' to phase '2' for a background flow,

(34e) corresponds to the event of transition from Erlang-2 phase '1' to phase '2' for a foreground flow,

(34f) corresponds to the event of a background flow completion,

(34g) corresponds to the event of a foreground flow completion,

(34h) corresponds to the event of a MMPP state transition (the transitions can be found in Table 1) and

(34i),(34j) correspond with the dummy transitions.

2.7 Experiments and results

Using the MDP models experiments have been set up that relate to the research questions about burstiness and different file size distributions. Before the experiments with the previously defined MDP's are conducted a term will be introduced for describing the resulting policies.

Switching curve Given two nodes and the number of flows on each node n_1, n_2. The policy is a function on the observed number of flows n_1, n_2. In the policies for the MDP's a curve can be observed on which the selection of a specific node changes to the other node. This curve will be denoted by the term 'switching curve'.

2.7.1 Poisson arrivals with exponential file sizes

Experiment In the first experiment the model for Poisson arrivals with exponential file sizes will be examined. Many experiments have been done on the model for Poisson arrivals and exponential file sizes. One experiment is included in this thesis to illustrate the impact of λ_0 on the optimal policy. The experiment has parameters:

- $\lambda_1 = .3, \lambda_2 = .7$,
- $\mu_0 = \mu_1 = \mu_2 = 1$,
- $C_1 = C_2 = 1$.

Here the arrival intensity for background traffic is chossen asymmetrically with rates λ_1 and λ_2 for background traffic on node 1 and 2. Furthermore the file sizes for both foreground and background flows are equal and also the capacities for both nodes C_1 and C_2 have been chosen equal. The foreground arrival rate is varied from low to high intensiy within the following parameter set: λ_0 was chosen $\{0.01, 0.05, 0.1, 0.2, 0.25\}$. The idea of this experiment is to find out the impact of foreground traffic on the policy for nodes with equal capacity but with different (background) loads.

Results When λ_0 becomes large the MDP gave unexpected results. First the MDP had difficulties in converging therefore an aperiodicity transformation had to be applied. Secondly the policy provides decisions that intuitively seem not to be optimal. In this case the switching curve is for some values on the heaviest loaded node side, while intuitively it should be on the lowest loaded node side. Therefore different policies have been examined for a fixed set of parameters for the Poisson arrival model with exponential file sizes. The choice of fixed parameters was:

- $\lambda_1 = .3, \lambda_2 = .7$,
- $\mu_0 = \mu_1 = \mu_2 = 1$,
- $C_1 = C_2 = 1$.

λ_0 was chosen $\{0.01, 0.05, 0.1, 0.2, 0.25\}$ For the different λ_0 the MDP came with policies:

- Red choose node 1
- Green choose node 2
- $\downarrow n_1$, # flows node 1
- $\rightarrow n_2$, # flows node 2

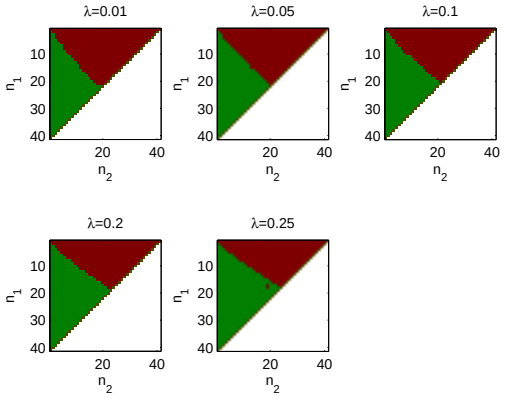


Figure 11: Policies for different λ_0.

Each of these policies has been examined for MDP's with different λ_0 as foreground arrival rate. In Table 3 the row dimension corresponds to the policies calculated using rate λ_0 as foreground flow arrival rate. The column dimension corresponds to the λ_0 used for calculation of the expected sojourn time while using the policy that was calculated using the λ_0 in the corresponding row dimension. The difference is calculated by taking the policy with the lowest expected sojourn time $\mathbb{E}[S|\lambda_{min}]$ and dividing the differences for all policies $\mathbb{E}[S|\lambda_0] - \mathbb{E}[S|\lambda_{min}]$ by $\mathbb{E}[S|\lambda_{min}]$:

$$\frac{\mathbb{E}[S|\lambda_0] - \mathbb{E}[S|\lambda_{min}]}{\mathbb{E}[S|\lambda_{min}]} \times 100\%. \tag{35}$$

Table 3: Difference in average sojourn time for policies obtained with λ_0 from the row and expected sojourn time calculated using the λ_0 in the column, with equal server capacity.

policy/λ_0	0.01	0.05	0.1	0.2	0.25
0.01	0.0000%	0.0187%	0.0987%	0.3626%	0.5566%
0.05	0.0195%	0.0000%	0.0183%	0.1242%	0.2288%
0.1	0.0375%	0.0074%	0.0000%	0.0000%	0.0200%
0.2	0.0486%	0.0209%	0.0144%	0.0005%	0.0000%
0.25	0.0826%	0.0583%	0.0524%	0.0239%	0.0072%

We observe that the case $\lambda_0 = 0.1$, although the switching curve is on the side of the server with most arrivals, the policy is still optimal. For higher λ_0 this is not the case, but the difference in average reward is really small.
Consider again the graphical representation of the policies and that background traffic node 2 has a higher arrival rate. Another observation that can be made is that when the foreground arrival rate becomes lower the switching curve moves to the node with low backtround traffic arrival intensity (node 1). (More decisions become red where red denotes the decision 'select node 2').

2.7.2 Varied burstiness with exponential file sizes

Experiment As described in Section 2.6.1 MMPP will be parameterized on burstines using a parametrization that is based on duty cycle (% of time on) D and average cycle period length T. The arrival rate of background flows will be assumed very low compared to the arrival rate of foreground flows. The duty cycle will be varied in the set $D \in \{100\%, 70\%, 50\%, 25\%, 10\%\}$ and the expected on-off cycle length will be varied in the set $T \in \{2.5, 10, 25\}$. This will create for each set-up $5 \times 3 = 15$ resulting policies for different burstiness in D and T. In this section six MMPP experiment set-ups are defined that will examine the impact of burstiness in arrivals. The first four experiment set-ups do consider:

- Poisson arrivals with rates $\lambda_0, \lambda_1, \lambda_2$,
- exponential filesizes $\mu_0 = \mu_1 = \mu_2 = 1$,
- two foreground flow arrival rates $\lambda_0 \in \{0.05, 0.01\}$,
- asymmetric background flow arrival rates $\lambda_1 = 0.1$, $\lambda_2 = 0.5$,
- asymmetric capacities for both nodes $C_1 = 1, C_2 = 1.5$ and $C_1 = 1, C_2 = 1.5$.

Note that the foreground flow arrival rates have been chosen low. The motivation of this choice can be found in the results of the experiments in 2.7. This experiments indicated that most impact in asymmetry in foreground flow arrival rates can be found in combination with low foreground flow arrival rates. For the first four MMPP experiments the node with the highest background flow arrival rate has always the highest load. Therefore the last two experiment set-ups are based on a scenario where the node with highest capacity has the

lowest load. Therefore the following parameters have been changed with respect to the first four set-ups:

- asymmetric background flow arrival rates $\lambda_1 = 0.1$, $\lambda_2 = 0.2$,
- asymmetric capacities for both nodes $C_1 = 1, C_2 = 3$ and $C_1 = 1, C_2 = 3$.

The six experiment set-ups can be summarized by:

Set-up 1: $\lambda_0 = 0.05, \lambda_1 = 0.1, \lambda_2 = 0.5, C_1 = 1.5, c_2 = 1$,

Set-up 2: $\lambda_0 = 0.05, \lambda_1 = 0.1, \lambda_2 = 0.5, C_1 = 1, c_2 = 1.5$,

Set-up 3: $\lambda_0 = 0.01, \lambda_1 = 0.1, \lambda_2 = 0.5, C_1 = 1.5, c_2 = 1$,

Set-up 4: $\lambda_0 = 0.01, \lambda_1 = 0.1, \lambda_2 = 0.5, C_1 = 1, c_2 = 1.5$,

Set-up 5: $\lambda_0 = 0.01, \lambda_1 = 0.1, \lambda_2 = 0.2, C_1 = 3, C_2 = 1$,

Set-up 6: $\lambda_0 = 0.01, \lambda_1 = 0.1, \lambda_2 = 0.2, C_1 = 3, C_2 = 1$.

Results The MMPP experiments consisted of examining the impact of burstiness on the server selection policy. Because for equal capacity the 'switching curve' remains close to the diagonal $n_1 = n_2$ for MMPP the experiments have been calculated using asymmetric capacities. Consider the model with MMPP arrivals and exponential file sizes (See Section 2.6.2 equation (32)). For all experiments the average file sizes have been chosen equal for all arrivals, $\mu_0 = \mu_1 = \mu_2 = 1$. For the nodes two combinations of asymmetric capacities are considered: $C_1 : C_2 = 1 : 1.5$ and $C_1 : C_2 = 1.5 : 1$. Furthermore, the arrival rate for background traffic on node 1 is chosen $\lambda_1 = 0.1$ and $\lambda_2 = 0.5$ for node 2. The foreground arrival rate is $\lambda_0 = 0.05$. As described in Section 2.6.1 burstiness of traffic can be expressed in two terms:

D the duty cycle D is varied in the range $\{100\%, 70\%, 50\%, 25\%, 10\%\}$,

T the cycle time is varied in the range $\{2.5, 10, 25\}$.

Note that when the duty cycle is 100% the MMPP model is equal to the models with Poisson arrivals. Figures 12 and 13 contain a graphical representation of the policies calculated using the MDP models. In these graphs each color corresponds to a server selection decision:

red select node 1 on foreground file arrival,

green select node 2 on foreground file arrival.

For the impact of burstiness the policies for the bursty arrival processes are compared with the Poisson arrival processes. The differences between the Poisson and bursty processes are coded with bright colors:

pink denotes for bursty model: select node 1 while for Poisson model: select node 2,

yellow denotes for bursty model: select node 2 while for Poisson model: select node 1.

In short terms: bright expresses that the node selection changed to the other node.

The policies are represented as two-dimensional images where each dimension represents the number of flows on a node. The origin with both $n_1 = 0$ and $n_2 = 0$ is in the top left corner:

$\downarrow := n_1 = \#$ flows node 1

$\rightarrow := n_2 = \#$ flows node 2

First the policies are presented with foreground traffic arrival rate $\lambda_0 = 0.05$. In Figures 12 and 13, T increases vertically and D decreases horizontally. Thus the bottom right corner is most bursty.

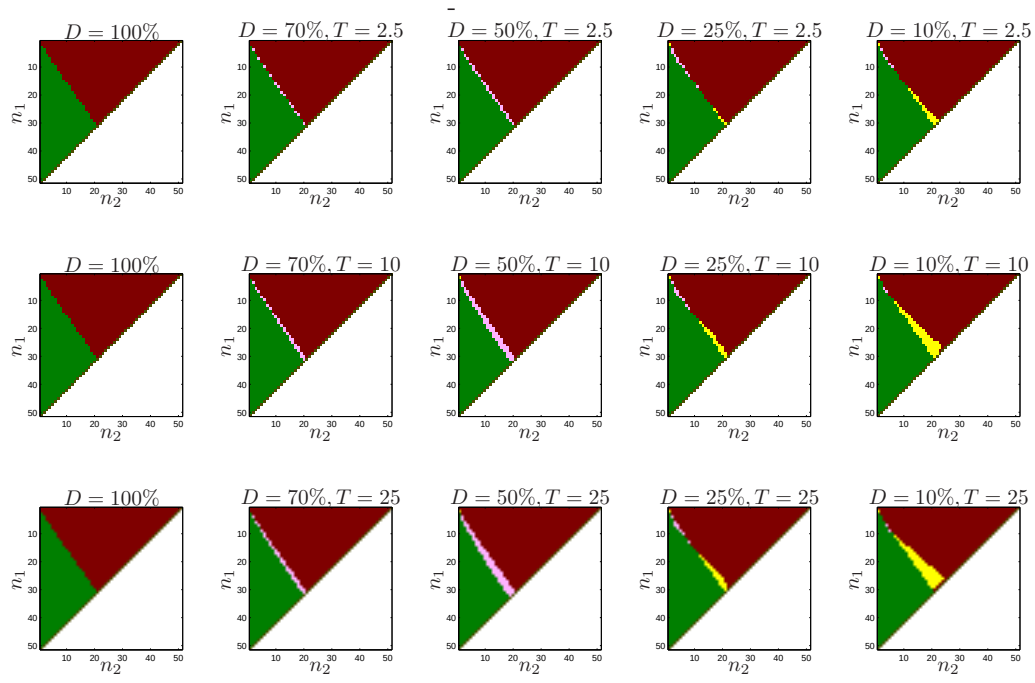


Figure 12: Optimal policies for different values of D on the horizontal axis and T on the vertical axis, with $\lambda_0 = 0.05$, $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $C_1 = 1.5$, $C_2 = 1$.

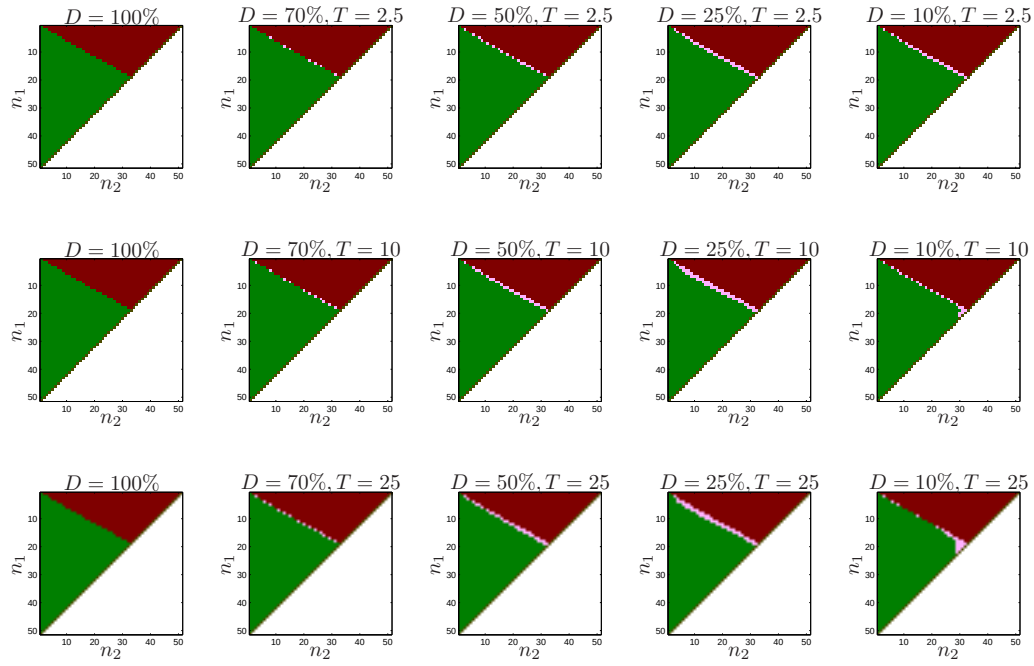


Figure 13: Optimal policies for different values of D on the horizontal axis and T on the vertical axis, with $\lambda_0 = 0.05$, $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $C_1 = 1$, $C_2 = 1.5$.

Remember that the arrival rate to node 2 is higher than to node 1. What can be observed is that for both asymmetrical cases $C_1 : C_2 = 1 : 1.5$ and $C_1 : C_2 = 1 : 1.5$ when T grows the 'switching curve' moves to the node with low background arrival rate (green area). For $C_1 : C_2 = 1 : 1.5$ the 'switching curve' moves to the server with low with low background arrival rate until D is decreased to 50%. For $C_1 : C_2 = 1 : 1.5$ the 'switching curve' moves to the server with low with low background arrival rate until D is decreased to 25%. This gives the impression that the server becomes to unstable when the burstiness high in terms of duty cycle. This can be clarified by the fact that for the model the arrival rate in the burstiness model is inversely proportional to the duty cycle:

$$\lambda_H = \frac{\lambda}{D}$$

With a duty cycle of 10% the arrival intensity of a burst can be really high: $\lambda_H = 10\lambda$.
This raises the question what will happen when the arrival intensity of the foreground traffic becomes really low. Therefore, the policies have also been calculated for $\lambda_0 = 0.01$. The resulting policies can be found in Figures 14 and 15.

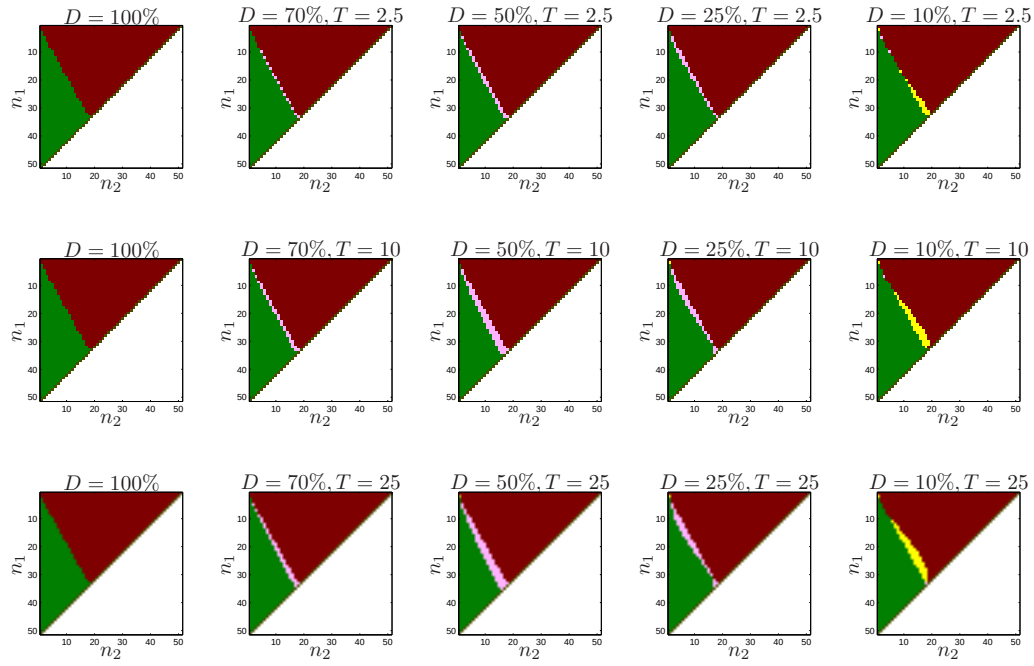


Figure 14: Optimal policies for different values of D on the horizontal axis and T on the vertical axis, with $\lambda_0 = 0.01$, $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $C_1 = 1.5$, $C_2 = 1$.

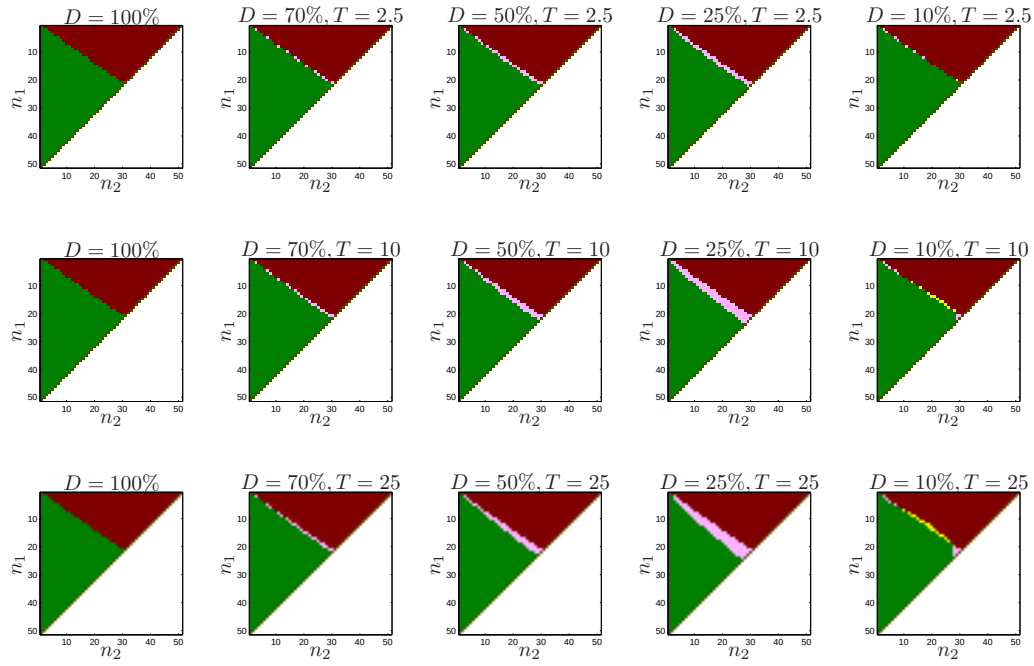


Figure 15: Optimal policies for different values of D on the horizontal axis and T on the vertical axis, with $\lambda_0 = 0.01$, $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $C_1 = 1$, $C_2 = 1.5$.

In Figures 14 and 15 it can be observed that when the duty cycle decreases D and the cycle time increases that the 'switching curve' will move the node with lower background traffic arrival intensity. For these parameters the 'switching curve' will move to the lower arrival intensity server until a duty cycle of 10%. This is lower than the 25% of the case with $\lambda_0 = 0.05$ and only the case when the node with high arrival rate $\lambda_2 = 5$ has the lowest capacity $C_2 = 1$. The difference between the Poisson arrival policies (on the left) and the bursty arrivals is larger than for in policies with $\lambda_0 = 0.05$. Now is is interesting to compare the difference for the policies for $\lambda_0 = 0.05$ and $\lambda_0 = 0.01$. In Figures 16 and 17 the policies for $\lambda_0 = 0.05$ are displayed as a difference with the policies for $\lambda_0 = 0.01$. Here:

red denotes that both for $\lambda_0 = 0.05$ and $\lambda_0 = 0.01$ the policies select node 1 for a new foreground arrival,

green denotes that both for $\lambda_0 = 0.05$ and $\lambda_0 = 0.01$ the policies select node 2 for a new foreground arrival,

pink denotes that for $\lambda_0 = 0.05$ the policy selects node 1 but for $\lambda_0 = 0.01$ the policy selects node 2,

yellow denotes that for $\lambda_0 = 0.05$ the policy selects node 2 but for $\lambda_0 = 0.01$ the policy selects node 1.

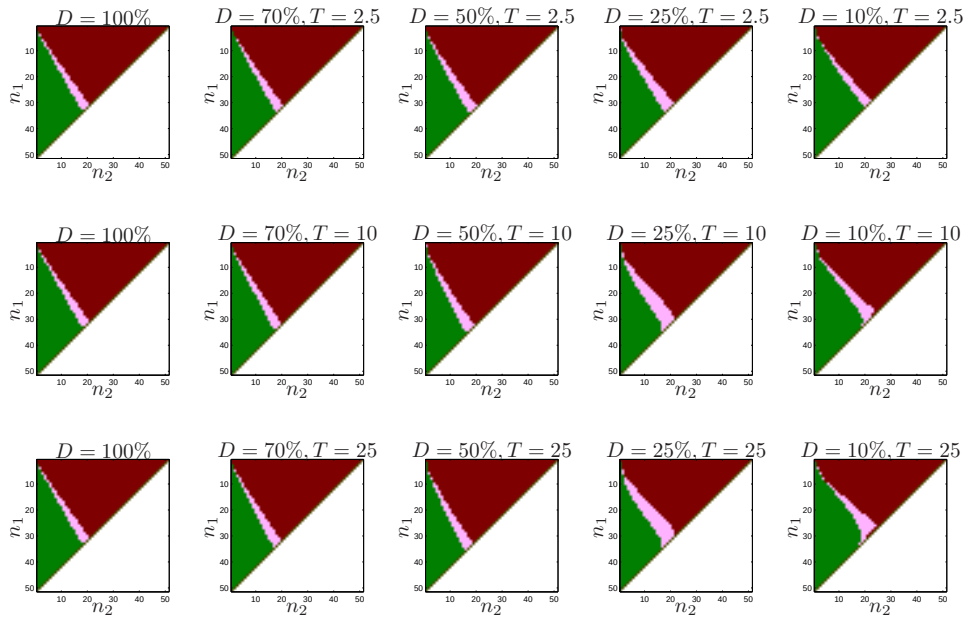


Figure 16: Difference between optimal policies with $\lambda_0 = 0.05$ and $\lambda_0 = 0.01$ for different values of D on the horizontal axis and T on the vertical axis and $C_1 : C_2 = 1.5 : 1$.

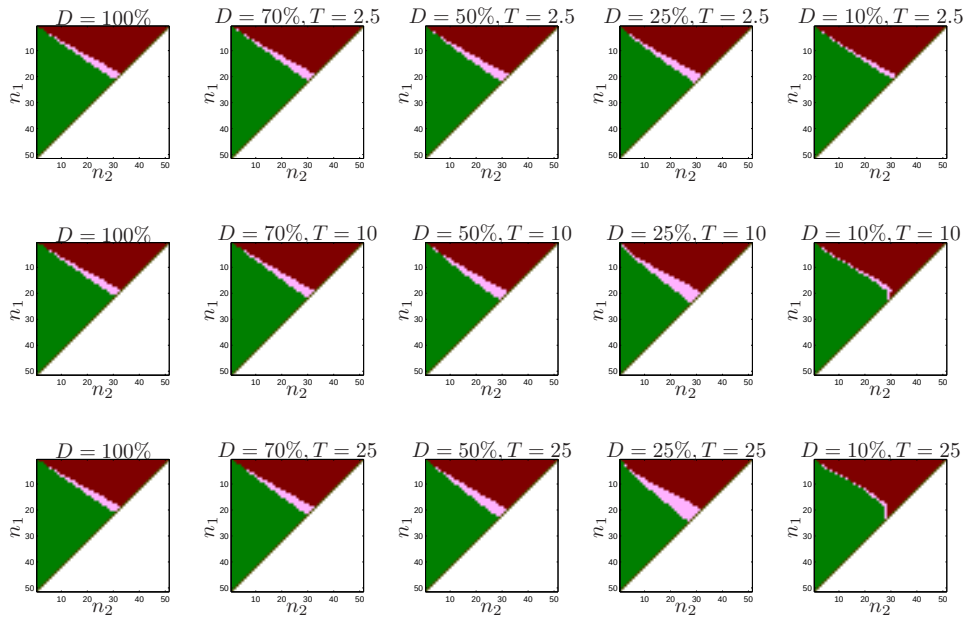


Figure 17: Difference between optimal policies with $\lambda_0 = 0.05$ and $\lambda_0 = 0.01$ for different values of D on the horizontal axis and T on the vertical axis and $C_1 : C_2 = 1 : 1.5$.

We observe that when λ_0 becomes smaller the impact of burstiness on the optimal policy becomes larger. With λ_0 lower the 'switching curve' moves to the server with lowest foreground arrival rate for both $C_1 : C_2 = 1 : 1.5$ and $C_1 : C_2 = 1.5 : 1$. The last observation can be explained:

$$\begin{array}{rcccl} C_1 : C_2 & = & 1.5 : 1 & : & \rho_1 = \frac{\lambda_1}{C_1} = \frac{0.1}{1.5} = \frac{2}{30}, \quad \rho_2 = \frac{\lambda_2}{C_2} = \frac{0.5}{1} = \frac{15}{30}, \\ C_1 : C_2 & = & 1 : 1.5 & : & \rho_1 = \frac{\lambda_1}{C_1} = \frac{0.1}{1} = \frac{3}{30}, \quad \rho_2 = \frac{\lambda_2}{C_2} = \frac{0.5}{1.5} = \frac{10}{30}. \end{array}$$

The background load for node 2 is always higher and so when traffic becomes more bursty or the arrival intensity of foreground traffic decreases the 'switching curve' moves to node 2. Of course this brings the question what will happen with the policies when the server with highest capacity has always the lowest load. Therefore the following parameters were selected: Server capacities $C_1 : C_2 = 1 : 3$ and $C_1 : C_2 = 3 : 1$, foreground traffic arrival rate $\lambda_0 0.01$, background traffic arrival rates $\lambda_1 = 0.3$ for node 1 and $\lambda_1 = 0.6$ for node 2. This results in loads:

$$\begin{array}{rcccl} C_1 : C_2 & = & 3 : 1 & : & \rho_1 = \frac{\lambda_1}{C_1} = \frac{0.3}{3} = \frac{1}{10}, \quad \rho_2 = \frac{\lambda_2}{C_2} = \frac{0.6}{1} = \frac{6}{10}, \\ C_1 : C_2 & = & 1 : 3 & : & \rho_1 = \frac{\lambda_1}{C_1} = \frac{0.3}{1} = \frac{3}{10}, \quad \rho_2 = \frac{\lambda_2}{C_2} = \frac{0.6}{3} = \frac{2}{10}. \end{array}$$

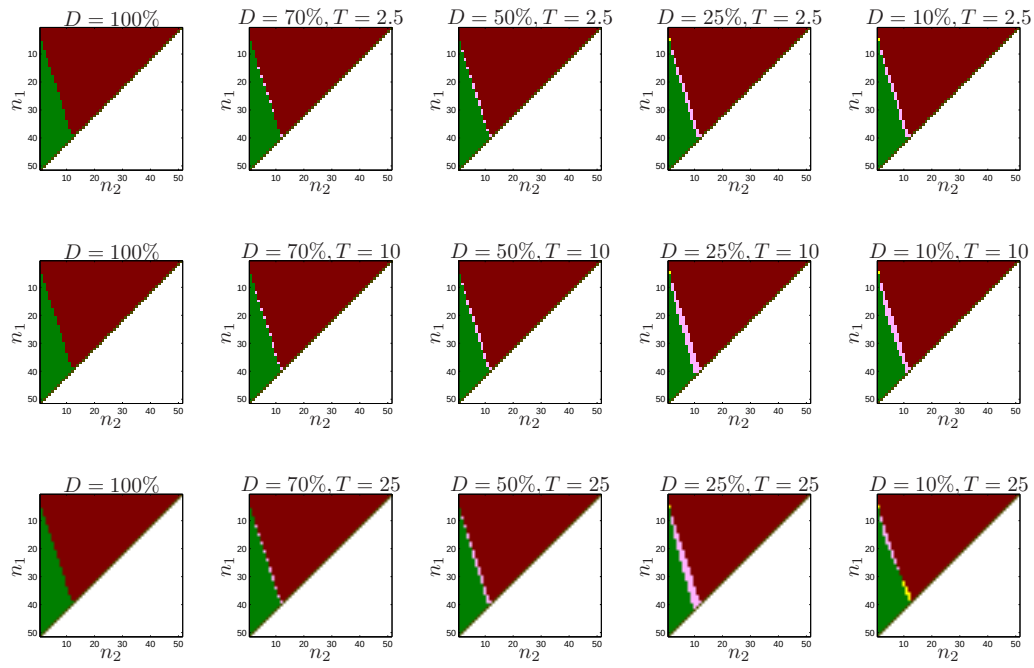


Figure 18: Optimal policies for different values of D on the horizontal axis and T on the vertical axis, with $\lambda_0 = 0.01$, $\lambda_1 = 0.1$, $\lambda_2 = 0.2$, $C_1 = 3$, $C_2 = 1$.

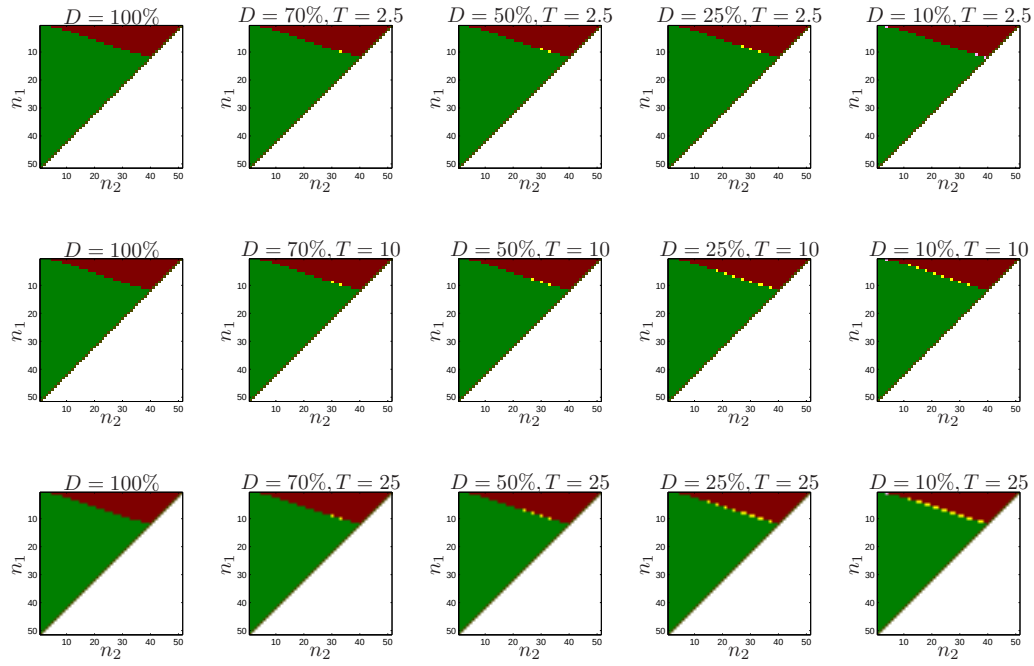


Figure 19: Optimal policies for different values of D on the horizontal axis and T on the vertical axis, with $\lambda_0 = 0.01$, $\lambda_1 = 0.1$, $\lambda_2 = 0.2$, $C_1 = 1$, $C_2 = 3$.

Figures 18 and 19 indeed show that when the burstiness increases the 'switching curve' moves to the server with lowest background load. For Figure 19 this is hard to observe due to relative small difference in background traffic load ($\rho_1 : \rho_2 = 3 : 2$) but the yellow spots indicate a slight movement of the 'switching curve' to node 2 which has in that case the lowest load.

2.7.3 Impact of different file size distributions

Experiment Besides burstiness it is also interesting to know the impact of the file size distribution on the optimal policy. Therefore the experiment set-ups will be varied on duty cycle $D \in \{100\%, 70\%, 50\%, 25\%, 10\%\}$ and different distributions $E_2, Exp, H_{2,c^2=2}, H_{2,c^2=4}, H_{2,c^2=16}$ sorted on the squared coefficients of variation $c^2 \in \{0.5, 1, 2, 4, 16\}$. This will generate a grid of $5 \times 5 = 25$ policies in D and c^2. In thise experiments T is fixed to $T = 10$. The two experiment set-ups do consider:

- MMPP arrivals with rates $\lambda_0, \lambda_1, \lambda_2$,
- E_2, exponentially and H_2 distributed filesizes with expectations $\frac{1}{\mu_0} = \frac{1}{\mu_1} = \frac{1}{\mu_2} = 1$,
- foreground flow arrival rate $\lambda_0 = 0.01$,
- asymmetric background flow arrival rates $\lambda_1 = 0.1$, $\lambda_2 \in \{0.5, 0.8\}$,
- symmetric capacities for both nodes $C_1 = C_2 = 1$.

The list of parameters for the two parameter sets can be summarized by:

Set-up 1: $\lambda_0 = 0.01, \lambda_1 = 0.1, \lambda_2 = 0.5, C_1 = C_2 = 1,$

Set-up 2: $\lambda_0 = 0.01, \lambda_1 = 0.1, \lambda_2 = 0.8, C_1 = C_2 = 1,$

Results Another interesting question is: what is the impact of the filesize distribution on the optimal policy? Therefore the MDPs also have been examined with the Erlang and H_2-distributions with:

- Erlang squared coefficient of variation $c^2 = 0.5$,
- exponential squared coefficient of variation $c^2 = 1$ and
- hyper-exponential squared coefficients of variation $c^2 = \{2, 4, 16\}$.

For these experiments the parameters are:

- $\lambda_0 = 0.01$,
- $\lambda_1 = 0.1$,
- $\lambda_2 = 0.5$ for Figure 20 and 0.8 for Figure 21,
- $\mu_0 = \mu_1 = \mu_2 = 1$,
- $C_1 = C_2 = 1$,
- $T = 10$ (cycle time).

On the columns the c^2 is varied $c^2 = \{.5, 1, 2, 4, 16\}$. On the rows the duty cycle is varied $D = \{100\%, 70\%, 50\%, 25\%, 10\%\}$.

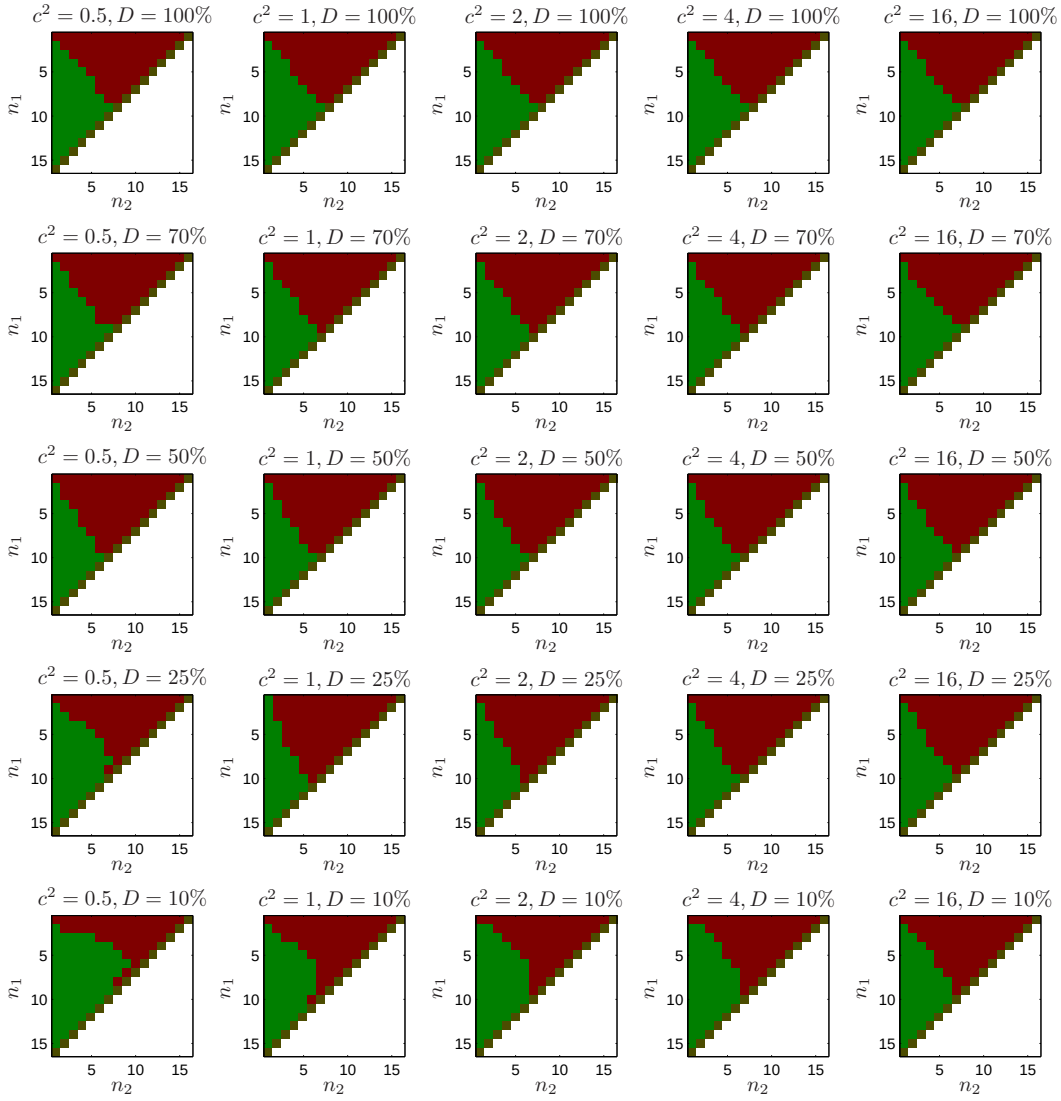


Figure 20: Impact of filesize distribution for different values of c^2 on the horizontal axis and burstiness values D on the vertical axis, with $\lambda_0 = 0.01$, $\lambda_1 = 0.1$, $\lambda_2 = 0.5$, $C_1 = 1$, $C_2 = 1$.

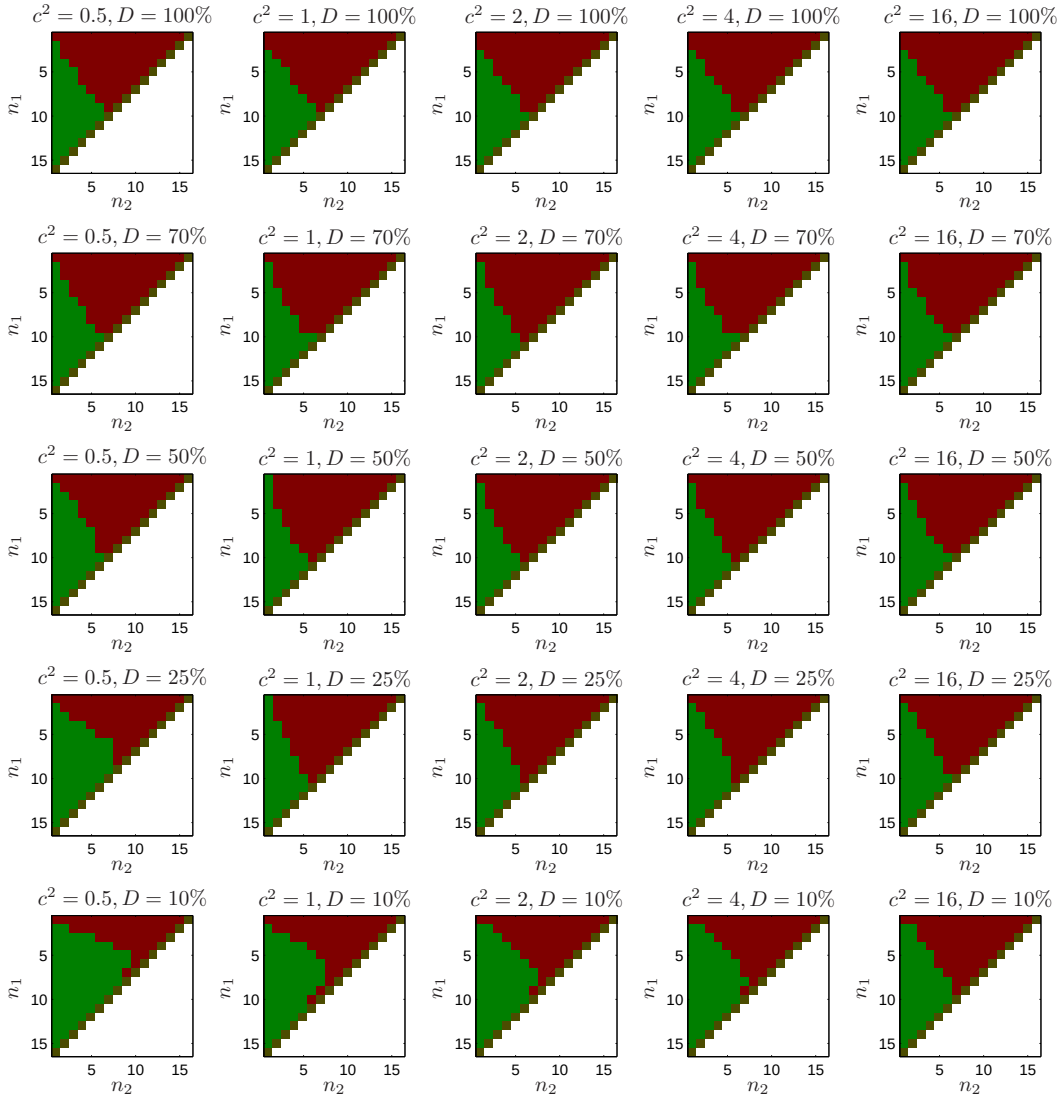


Figure 21: Impact of filesize distribution for different values of c^2 on the horizontal axis and burstiness values D on the vertical axis, with $\lambda_0 = 0.01$, $\lambda_1 = 0.1$, $\lambda_2 = 0.8$, $C_1 = 1$, $C_2 = 1$.

In Figures 20 and 21 can be observed that when the duty cycle is not really small, $D \geq 50\%$, the squared coefficient of variation c^2 does not really has a large impact on the policy. The differences do start in Figure 21 with the Erlang model with $D = 50\%$. For low D, $D < 50\%$ the policies with small c^2 on the bottom left start to change earlier than those with high c^2 bottom right. Although some differences can be observed it is not completely sure what is the impact of the state space limitation. In theory the state space is infinitly large. In order to enable computation the state space has to be limited to some large

number. This limitation should be large enough in order to minimize the probability the system grows larger than the limited state space. For more accurate results the limitation should be high. However for the complex models the state space grows such large that the limits of CPU (speed) and memory of the high end Personal Computer are reached.

2.7.4 Impact of difference in distribution between flows

Experiment Until now, in the experiments, the squared coefficent of variation has been chosen equal. The final experiment on MDPs will consider different squared coefficent of variation c^2 for foreground and background flows. In these experiments the filesize distribution will be limited to the H_2-distribution. The parameters for these experiments are:

- MMPP arrivals with rates $\lambda_0, \lambda_1, \lambda_2$,
- H_2 distributed filesizes with expectations $\beta_0 = \beta_1 = \beta_2 = 1$,
- foreground flow arrival rate $\lambda_0 = 0.01$,
- asymmetric background flow arrival rates $\lambda_1 = 0.1$, $\lambda_2 \in \{0.5, 0.8\}$,
- symmetric capacities for both nodes $C_1 = C_2 = 1$,
- asymmetric coefficients of variation $c_0^2 = 1$, $c_1^2, c_2^2 \in \{1, 2, 4, 16\}$.

This experiment set-up corresponds to the set-up in 2.7.3 in the way how the duty cycle D and squared coefficient of variation c^2 is varied. The experiment differs the set of file size distributions. Distributions are now limited to H_2. For foreground flows c^2 is varied in the set $\{1, 2, 4, 16\}$. Background flows will have a fixed $c_1^2 = c_1^2 = 1$. This will generate $5 \times 4 = 20$ policies varied over the squared coefficient of variation of foreground flows $c_0^2 \in \{1, 2, 4, 16\}$ and duty cycle for the arrivals $D \in \{100\%, 70\%, 50\%, 25\%, 10\%\}$.

Results Last experiment did compare the policies with for all file size distribution equal squared coefficient of variation. Consider c^2 comparison model parameters for Figure 21. Now the squared coefficient of variation of the background traffic is chosen $c_1^2 = c_2^2 = 1$. In Figure 22 on the columns the foreground traffic squared coefficient of variation is varied $c_0^2 = \{1, 2, 4, 16\}$. On the rows the duty cycle is varied $D = \{100\%, 70\%, 50\%, 25\%, 10\%\}$. In Figure 22 the comparison is done with the exponential file sizes for different D. So the comparison of the policies is with for each D the exponential file size policies on the left.

pink denotes that for exponential file sizes the policy selects node 1 but for other c_0^2 the policy selects node 2,

yellow denotes that for exponential file sizes the policy selects node 2 but for other c_0^2 the policy selects node 1.

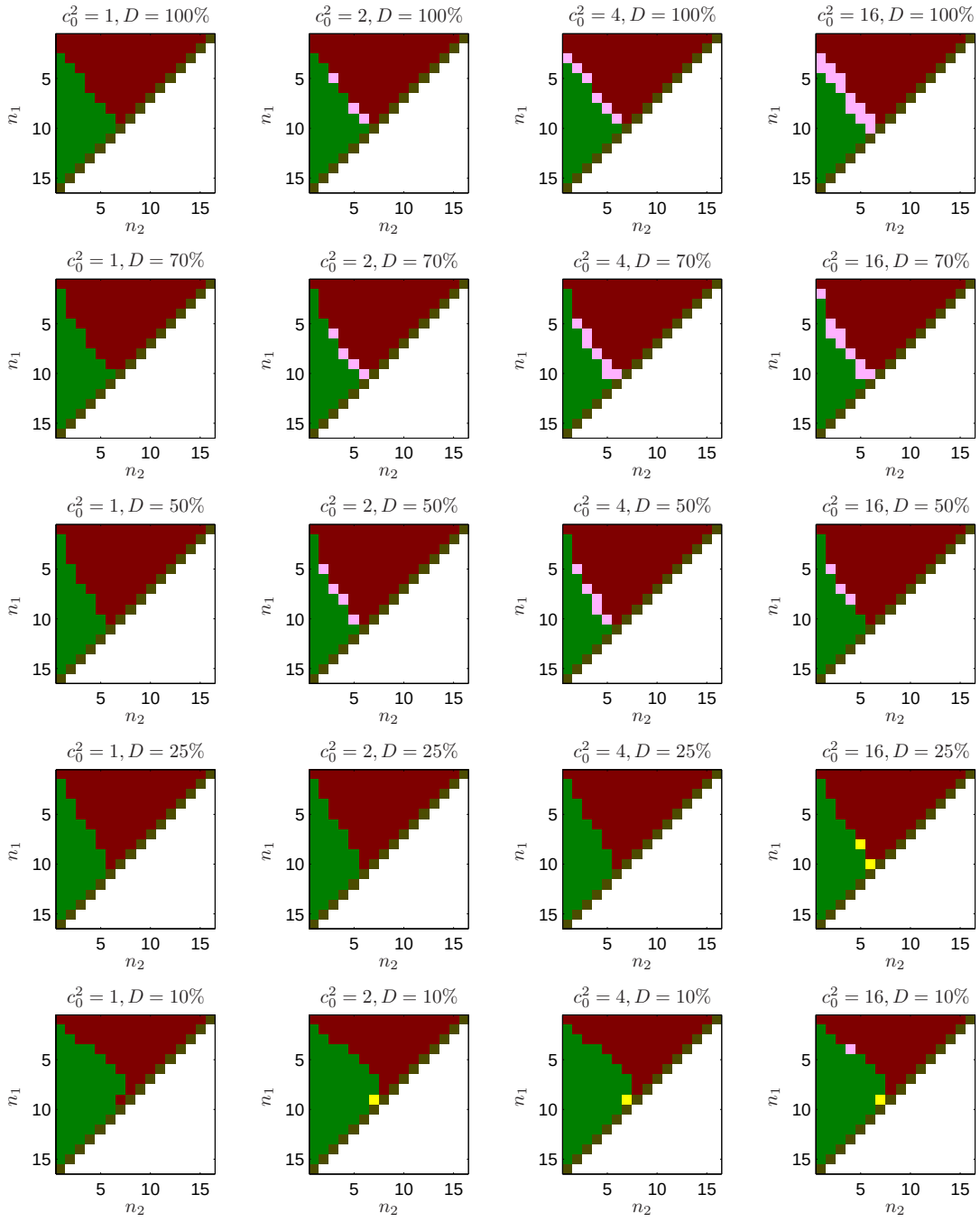


Figure 22: Impact of filesize distribution for different values of c_0^2 (foreground flows) on the horizontal axis and burstiness values D on the vertical axis, with $\lambda_0 = 0.01$, $\lambda_1 = 0.1$, $\lambda_2 = 0.8$, $C_1 = 1$, $C_2 = 1$.

What can be observed is that when the c_0^2 becomes larger the 'switching curve' moves to the node with the lowest arrival rate. This is the same behavior as earlier observed when the burstiness increases.

2.8 Conclusions

- When burstiness increases ($D \downarrow$ or $T \uparrow$) the preference for the lowest loaded node increases. This can be observed as the movement of the 'switching curve' to the direction of the lowest loaded node. This movement is bigger when the foreground traffic arrival intensity λ_0 increases.
- When λ_0 decreases the 'switching curve' moves to the lowest loaded server.
- The impact of the filesize distribution is small when the squared coefficient of variation c^2 is equal for both background and foreground files. This should however be examined by applying a larger statespace and assymmetric server capacities.
- When the squared coefficient of variation of foreground traffic c_0^2 grows larger, with fixed background squared coefficient of variation $c_1^2 = c_2^2 < c_0^2$, the switching curve moves to the lowest loaded node.

3 Dynamic splitting using Bayesian dynamic programming

In the MDP models of Chapter 2, the difference between foreground and background traffic can not be observed. This is a partial observation problem. In Section 2.3 a partial observation approach is defined that is based on conditioning over the time limiting distribution. Using this approach information that can be found in the sequence of observations remains unused. This 'observation sequence information' provides a potential for improving the performance for server selection models under the partial observation assumption.

For example consider the model with Poisson arrivals and exponentially distributed file sizes in Section 2.4. If in an empty system only foreground files arrive then we know for sure that all observed flows are foreground traffic.

Bayesian dynamic programming enables the use of information that can be found in the sequence of observations. In the example we know with probability 1 that the system is in a state with only foreground flows. By using Bayesian dynamic programmic the information in the sequence of observations can be used to improve the optimal strategies of Chapter 2. In this chapter first the framework for Bayesian dynamic programming will be formulated in Section 3.1. The problem of partial observation that was introduced in Section 2.3 will in Section 3.2 be formulated as a Bayesian dynamic programming problem. A drawback of Bayesian dynamic programming is the size of the solution (due to large statespace). Solutions for reducing the size for this Bayesian dynamic programming problem will be studied in Section 3.3. In Section 3.4 the structure of transitions between states will be examined using a (simple) simulation program. To overcome the difficulty of statespace complexity, in Section 3.5 Bayesian information will be combined with the solution of the full observable MDP defined in Equation eqrefbasic MDP in Section 2.2. The performance results of the combination of Bayesian information with a fully observable MDP will be discussed in Chapter 5.

3.1 Framework

For the Bayesian dynamic programming define:

$\mathcal{S}$ the state space of the system,

O observation space which contains all observations that can be made,

$\mathcal{A}$ the set of possible actions or decisions,

$\mathcal{P} = \left\{ u \in [0,1]^{|\mathcal{S}|} \mid \sum_{s \in \mathcal{S}} u(s) = 1 \right\}$ the information state or Bayesian belief distribution on $\mathcal{S}$.

Note that in the Bayesian dynamic programming decisions are based on the information state (which is in $\mathcal{P}$). When a system is in a state $s \in \mathcal{S}$ sometimes it is possible that this can lead to multiple observations. Therefore $q(s,o)$ is introduced which is the probability of observing $o \in O$ when the system is in

state $s \in \mathcal{S}$. For the server selection models it is always the case that when the system is in a certain state it can only result in one observation. In fact for the model with Poisson arrivals and exponential file sizes the observation corresponding to a given state $(b_1, f_1, b_2, f_2) \in \mathcal{S}$ is defined by Equation (6):

$$o\left(b_1, f_1, b_2, f_2\right) = \left(b_1 + f_1, b_2 + f_2\right),$$

with b_i the number of background flows on node i and f_i the number of foreground flows on node i. This implies that

$$q\left(s, o\right) = \mathbb{1}\left[f_1 + b_1 = o_1 \wedge f_2 + b_2 = o_2\right]$$

where $\mathbb{1}$ is the indicator funcion. With Bayesian dynamic programming the information state is updated for each observation $o \in O$ with:

$u\left(s\right)$ $u \in \mathcal{P}$, is prior belief the system is in state s,

$v\left(s\right)$ $v \in \mathcal{P}$, is posterior belief the system is in state s in which the information obtained from the last observation is embedded.

In the partial information approach state transitions can be defined in three parts:

1 What transitions can occur from information state u to information state v. In the information state transition the information about the last action $a \in \mathcal{A}$ and observation $o \in O$ is added to prior information state u. Information state v contains all information about the past including the last observation and action. v can be calculated using the Bayesian update rule. This is the actual transition from prior information state u to the new posterior information state v. The Bayesian update is defined by:

$$\begin{aligned} v\left(s'\right) &= \mathbb{P}\left(s' | u, a, o\right) \\ &= \begin{cases} \dfrac{\sum\limits_{s \in \mathcal{S}} u\left(s\right) p\left(s, a, s'\right) q\left(s', o\right)}{\sum\limits_{s \in \mathcal{S}} \sum\limits_{s' \in \mathcal{S}} u\left(s\right) p\left(s, a, s'\right) q\left(s', o\right)} & \text{for} \quad \sum\limits_{s \in \mathcal{S}} \sum\limits_{s' \in \mathcal{S}} u\left(s\right) p\left(s, a, s'\right) q\left(s', o\right) > 0 \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{36}$$

2 What is the transition probability from information state u to information state v. For the original fully observable MDP the transition probabilities are known as $p\left(s, a, s'\right)$. From this transition probabilities the information state transition probabilities $p'\left(u, a, v\right)$ can be derived:

$$p'\left(u, a, v\right) = \sum_{s \in \mathcal{S}S} \sum_{s' \in \mathcal{S}} b\left(s\right) p\left(s, a, s'\right) q\left(s', o\right). \tag{37}$$

3 For the classical MDP to each state $s \in \mathcal{S}$ and decision $a \in \mathcal{A}$ a reward function $r\left(s, a\right)$ is defined. The information state reward function $r'\left(u, a\right)$ can be written in terms of $r\left(s, a\right)$:

$$r'\left(u, a\right) = \sum_{s \in \mathcal{S}} u\left(s\right) r\left(s, a\right) q\left(s, o\right). \tag{38}$$

With the information state transitions and probabilities defined partial information can be fit in the MDP value iteration equations:

$$\begin{aligned} V_{(t+1)\tau}(u) &= \\ \max_{a\in\mathcal{A}} &\left\{ r'(u,a)\,\tau + q'(u,a) \sum_{v\in\mathcal{P}} p'(u,a,v)\, V_{t\tau}(v) + (1-q'(u,a))\, V_{t\tau}(u) \right\} , \\ r'(u,a) &= \sum_{s\in\mathcal{S}} u(s)\, r(s,a)\, q(s,o) , \\ p'(u,a,v) &= \sum_{s\in\mathcal{S}} \sum_{s'\in\mathcal{S}} u(s)\, p(s,a,s')\, q(s',o) , \\ q'(u,a) &= \sum_{s\in\mathcal{S}} u(s)\, q'(s,a) . \end{aligned} \tag{39}$$

(We refer to [8] for the existence of deterministic generalized Markov policies for non-stationary Bayesian dynamic decision models.) Because the function q is already in use, $q'(s,a)$ is defined the geometric distribution parameter that enables the backward recursion by adding dummy transitions as described in Appendix 8.A.2 and τ is the expected time between the recursion steps. (See Section 8.A.2 for the value iteration definition).

3.2 Bayesian dynamic programming formulation

As described in the start of this chapter the idea is to apply Bayesian dynamic programming to the Poisson arrivals with exponential file sizes model in Section 2.4. This model has state space $(b_1, f_1, b_2, f_2) \in \mathcal{S}$ with b_i the number of background flows and f_i the number of foreground flows on node $i = 1, 2$. For the server selection partial information approach the observation space is defined as:

$$o = (o_1, o_2, o_3) \in O = {\mathbb{N}_0}^2 \times \{f, b\} .$$

$o_i = b_i + f_i$ corresponds to the number of flows on node $i \in \{1, 2\}$. The observation of the number of flows is not enough. The type of arrival does have impact on the information state update. For example, when a foreground arrival occurs, the system has certainly at least 1 foreground flow. With a background flow arrival the system can have no foreground flows at all. Therefore o_3 is added where $o_3 = f$ if the last arrival was foreground or $o_3 = b$ if the last arrival was foreground. For each observation (o_1, o_2, o_3) the information state can be updated using the Bayesian update. There are six possible observations with each their own information state update:

$(o_1, o_2) \to (o_1 + 1, o_2, b)$ a background flow arrival on node 1,

$(o_1, o_2) \to (o_1 + 1, o_2, f)$ a foreground flow arrival routed to node 1,

$(o_1, o_2) \to (o_1 - 1, o_2)$ a flow completion on node 1,

$(o_1, o_2) \to (o_1, o_2 + 1, b)$ a background flow arrival on node 2,

$(o_1, o_2) \to (o_1, o_2 + 1, f)$ a foreground flow arrival routed to node 2,

$(o_1, o_2) \to (o_1, o_2 - 1)$ a flow completion on node 2.

An information state is a distribution over the state space. However, the distribution on node 1 between background and foreground flows does not depend

on the distribution between background and foreground flows on node 2. For example when an arrival or completion occurs on node i this will not affect the belief on the distribution between foreground and background flows on node 2. Therefore the information state u can be split in two information states u_1 and u_2, one for each node. When for a node i o_i flows are observed, the information state only needs $o_i + 1$ entries. These are the number of combinations of foreground and background flows that will add up to o_i: $b_1 + f_i$. Because all entries add up to o_i the information state distribution for each node u_i can be defined as a probability distribution the number of foreground flows. The compact information state distribution for node i is now defined as

$$u_{io_i} \in \left\{ v \in [0,1]^{o_i} \mid \sum_{j=0}^{o_i} v(j) = 1 \right\}. \tag{40}$$

Here o_i is the observed number of flows that correspond to that distribution. With the information states defined the Bayesian update rules can be defined. Define:

$v_{i[o_i+1,f]}$ the resulting information state distribution for node i after a foreground flow arrival,

$v_{i[o_i+1,b]}$ the resulting information state distribution for node i after a background flow arrival,

$v_{i[o_i-1]}$ the resulting information state distribution for node i after a flow departure.

Then the update rules are:

$$\begin{aligned}
v_{1[o_1+1,f]}(i) &= \begin{cases} 0 & \text{if} \quad i = 0 \\ u_{1o_1}(i-1) & \text{if} \quad 0 < i \leq o_1 + 1 \end{cases} \\
v_{1[o_1+1,b]}(i) &= \begin{cases} u_{1o_1}(i) & \text{if} \quad 0 \leq i < o_1 + 1 \\ 0 & \text{if} \quad i = o_1 + 1 \end{cases} \\
v_{2[o_2+1,f]}(i) &= \begin{cases} 0 & \text{if} \quad i = 0 \\ u_{2o_2}(i-1) & \text{if} \quad 0 < i \leq o_2 + 1 \end{cases} \\
v_{2[o_2+1,b]}(i) &= \begin{cases} u_{2o_2}(i) & \text{if} \quad 0 \leq i < o_2 + 1 \\ 0 & \text{if} \quad i = o_2 + 1 \end{cases} \\
v_{1[o_1-1]}(i) &= u_{1o_1}(i+1)\tfrac{i+1}{o_1} + u_{1o_1}(i)\tfrac{o_1-i}{o_1}, \\
v_{2[o_2-1]}(i) &= u_{2o_2}(i+1)\tfrac{i+1}{o_2} + u_{2o_2}(i)\tfrac{o_2-i}{o_2}.
\end{aligned} \tag{41}$$

The observation transition probabilities are defined by:

$$\begin{aligned}
\mathbb{P}\left(u_{1o_1}, a, v_{1[o_1+1,o_2,f]}\right) &= \frac{\lambda_0 \mathbb{1}\left(a=0\right)}{\gamma}, \\
\mathbb{P}\left(u_{1o_1}, a, v_{1[o_1+1,o_2,b]}\right) &= \frac{\lambda_1}{\gamma}, \\
\mathbb{P}\left(u_{2o_2}, a, v_{2[o_1,o_2+1,f]}\right) &= \frac{\lambda_0 \mathbb{1}\left(a=1\right)}{\gamma}, \\
\mathbb{P}\left(u_{2o_2}, a, v_{2[o_1,o_2+1,b]}\right) &= \frac{\lambda_2}{\gamma}, \\
\mathbb{P}\left(u_{1o_1}, a, v_{1[o_1-1,o_2]}\right) &= \frac{\mu_1}{\gamma}, \\
\mathbb{P}\left(u_{2o_2}, a, v_{2[o_1,o_2-1]}\right) &= \frac{\mu_2}{\gamma}.
\end{aligned} \tag{42}$$

To enable an algorithm finding the optimal policy for the information states, discretization of the information states has to be applied. For the discretization scheme the interval $[0,1]$ is divided in $k+1$ linear discretization steps identified by $\frac{i}{k}$, $i = 0, \cdots, k$. From this discretized values a set of probability density functions, consisting of $o_i + 1$ probabilities, can be generated. The set of discretized distributions with $k+1$ discretization steps and o_i+1 probabilities will be denoted by $\mathcal{D}_{o_i,k}$. An example for such discretization scheme will provide some clarification:

$$\begin{aligned}
k &= 2, \\
o_i &= 0, 1, 2.
\end{aligned}$$

This will generate possible values: $\{0, .5, 1\}$. For the different z the set of discretized information states will be:

$$\mathcal{D}_{0,2} = \left\{1\right\}, \mathcal{D}_{1,2} = \begin{Bmatrix} 1 & .5 & 0 \\ 0 & .5 & 1 \end{Bmatrix}, \mathcal{D}_{2,2} = \begin{Bmatrix} 1 & .5 & .5 & 0 & 0 & 0 \\ 0 & .5 & 0 & .5 & 1 & 0 \\ 0 & 0 & .5 & .5 & 0 & 1 \end{Bmatrix}.$$

The posterior distribution v after an information state update will rarely be a distribution that exactly matches a distribution in the set of discretised distributions. Therefore the closest matching distribution has to be found. For comparison the squared error is used. For actual distribution p and the matched distribution $\hat{p}$ and the squared error is defined by:

$$ERR\left(p, \hat{p}\right) := \sum_i \left[p\left(i\right) - \hat{p}\left(i\right)\right]^2. \tag{43}$$

The squared error is minimized over the set of discretized distributions

$$\hat{v} = \min_{p \in \mathcal{D}_{o,k}} \sum_i \left[p\left(i\right) - v\left(i\right)\right]^2.$$

For the implementation one important implementation note can be made. When all the matching of distributions is executed on-line the algorithm will slow down

considerably. Therefore the matched information state transitions should be calculated off line. After implementing the MDP using information states only small state spaces could be examined ($\#jobs \leq 10$) due to memory and time limitations. Not only the solution requires a lot of memory off line, but it also requires a lot of memory on line.

3.3 State space reduction

When applying a solution based on information states a huge decision table should be stored on the traffic distribution node. Also the calculation of the optimal policy consumes lots of memory and CPU time. Therefore, simplification of the information state representation can be very useful. Consider the structure of information state updates.

When an idle server is observed, $o_i = (0)$ the number foreground and background flows can only be zero. The information state for that server will therefore be $u_{i0}\,(0) = 1$. Each arrival the type (foreground, background) can be observed and according to the policy the number of foreground and background flows can be tracked. This will be the case until a flow will be completed. When a flow leaves the system the type (foreground or background) can not be observed. The growth of uncertainty after a departure can also be found in the information state update equations. When foreground flows are admitted the information state distribution will shift to the right. If background flows arrive, the shape of the distribution will not change while the number of probabilities will increase by one. Figure 23 illustrates the event of a foreground flow arrival.

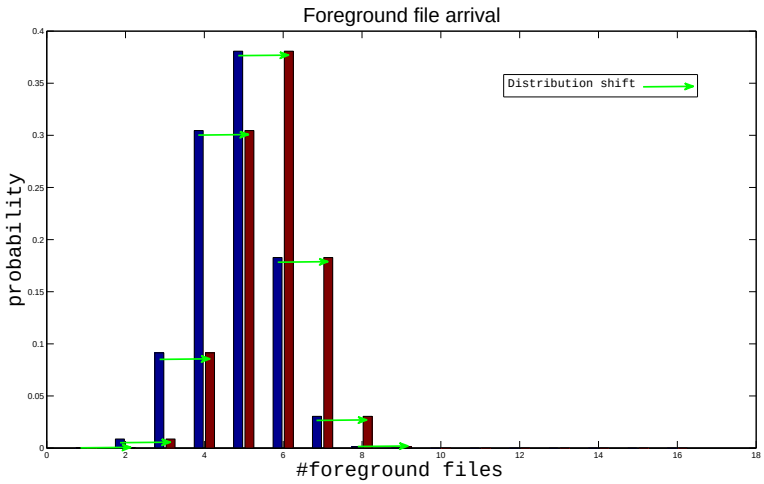


Figure 23: blue prior distribution, red posterior distribution.

In case of a foreground flow arrival the distribution (information state) will shift to the right.

In the example illustrated in Figure 24, for a number of flows the transmission will be completed.

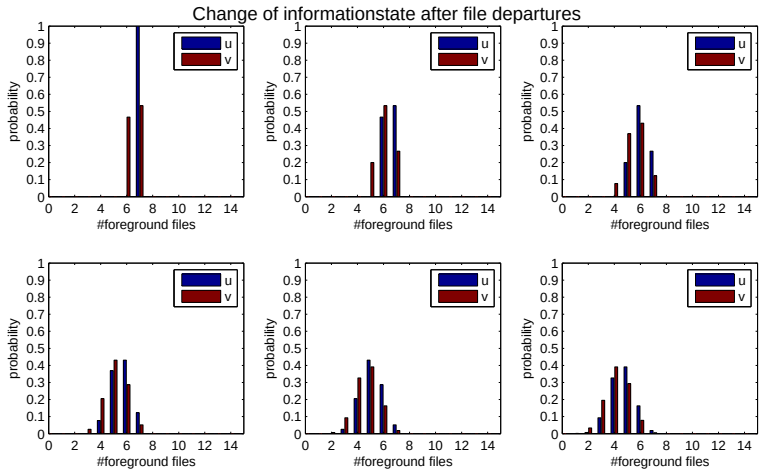


Figure 24: blue prior distribution, red posterior distribution.

After a series of departures the shape of a hyper-exponential distribution appears.

Assume that from an empty system n arrivals occurred and k flows are foreground type. Then the information state will be:

$$u = (u_0, \cdots, u_k, \cdots, u_n) = (0, \cdots, 1, \cdots, 0) \tag{44}$$

where $u(k) = 1$ and $u(i \neq k) = 0$. When the next events are only departures the information state structure will become:

i	n	$n-1$	$n-2$	$n-3$	$n-4$
$k-4$	0	0	0	0	$1\frac{k(k-1)(k-2)(k-3)}{n(n-1)(n-2)(n-3)}$
$k-3$	0	0	0	$1\frac{k(k-1)(k-2)}{n(n-1)(n-2)}$	$4\frac{k(k-1)(k-2)(n-k)}{n(n-1)(n-2)(n-3)}$
$k-2$	0	0	$1\frac{k(k-1)}{n(n-1)}$	$3\frac{k(k-1)(n-k)}{n(n-1)(n-2)}$	$6\frac{k(k-1)(n-k)(n-k-1)}{n(n-1)(n-2)(n-3)}$
$k-1$	0	$1\frac{k}{n}$	$2\frac{k(n-k)}{n(n-1)}$	$3\frac{k(n-k)(n-k-1)}{n(n-1)(n-2)}$	$4\frac{k(n-k)(n-k-1)(n-k-2)}{n(n-1)(n-2)(n-3)}$
k	1	$1\frac{n-k}{n}$	$1\frac{(n-k)(n-k-1)}{n(n-1)}$	$1\frac{(n-k)(n-k-1)(n-k-2)}{n(n-1)(n-2)}$	$1\frac{(n-k)(n-k-1)(n-k-2)(n-k-3)}{n(n-1)(n-2)(n-3)}$
$k+1$	0	0	0	0	0
$k+2$	0	0	0	0	0

From this development a structure can be recognized. First the scalars that precede the fractions are binomial coefficients $\binom{d}{i-(k-d)}$. This be clarified by the structure of the information state update equations. These have the structure $v(i) = c_1 u(i) + c_2 u(i+1)$ where c_1 and c_2 are the fractions as described in the information state update equations. These fractions c_1 and c_2 cause $c_1 u(i)$ and $c_2 u(i+1)$ to be equal up to a scalar. The scalar is the scalar as in the binomial coefficient. The binomial can be calculated in a series of summations:

$$
\begin{array}{ccccccccccc}
 & & & & 1 & & & & & & \\
 & & & 1 & + & 1 & & & & & \\
 & & 1 & + & 2 & + & 1 & & & & \\
 & 1 & + & 3 & + & 3 & + & 1 & & & \\
1 & + & 4 & + & 6 & + & 4 & + & 1. & &
\end{array}
$$

This is eqactly what happens in the update equations. Thus when the state is exactly known (n flows of which k foreground type) the information state after d departures will become:

$$u_d(i) = \binom{d}{i-k+d} \frac{\frac{k!}{i!}\frac{(n-k)!}{(n-d-i)!}}{\frac{n!}{(n-d)!}}. \tag{45}$$

When expanding the binomial coefficient the equation will become

$$u_d(i) = \frac{d!}{(i-k+d)!\,(k-i)!} \frac{\frac{k!}{i!}\frac{(n-k)!}{(n-d-i)!}}{\frac{n!}{(n-d)!}}, \tag{46}$$

which can be rewritten to

$$b_d(i) = \frac{\frac{k!}{i!(k-i)!}\frac{(n-k)!}{(n-d-i)!(i-k+d)!}}{\frac{n!}{(n-d)!d!}} = \frac{\binom{k}{k-i}\binom{n-k}{d-(k-i)}}{\binom{n}{d}}. \tag{47}$$

This is equal to the hypergeometric distribution $\mathbb{P}_{N,k,n}(x)$ with parameters

- $N = n$ total number of arrived flows after the system has been empty,
- $k = k$ the total number of foreground flows before completion,
- $n = d$ the number of completed flows,
- $x = k - i$ the number of foreground flows that completed since the first flow completion.

Given Figures 23 and 24 the idea arises that the information state distribution will remain a peak shaped distribution. With a peak shaped distribution a distribution with a similar shape like the binomial distribution or the hypergeometric distribution is denoted. This brought the idea for finding a conjugate distribution for the information states. A conjugate distribution is a distribution that after a Bayesian information update is applied will remain in the same distribution family, but with different parameters. More information can be found in [4]. A well known example is the Bernoulli trial with unknown success probability p. In that case the prior belief has a $Beta(\alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)}$ distribution with $\beta(\alpha, \beta)$ the beta function. The Bernoulli experiment can have two possible outcomes with each their corresponding Bayesian update:

- In case of a success the posterior distribution will be $\frac{x^{\alpha}(1-x)^{\beta-1}}{B(\alpha+1,\beta)}$ which is a $Beta(\alpha+1, \beta)$ distribution,

- In case of a failure the posterior distribution will be $\frac{x^{\alpha-1}(1-x)^{\beta}}{B(\alpha,\beta+1)}$ which is a $Beta\,(\alpha,\beta+1)$ distribution.

Corresponding to a success or failure either the α parameter or β parameter will be increased by 1. As mentioned before, if from an empty system a sequence of arrivals occurs and after the arrival sequence only departures occur, the information state distribution will be a hypergeometric distribution. When between the departures an arrival occurs the information state is not hyper-exponentially distributed anymore. This is caused by the fact that after a foreground arrival the distribution will shift to the right. When a foreground arrival occurs there is at least one foreground job in the system thus the probability there are zero foreground jobs in the system will become zero.

3.4 Simulation program

To examine the impact of distribution shifts during the information state updates a simulation program was written. This program is based on the solution of the fully observable MDP model with Poisson arrivals and exponential file sizes that is described in Section 2.2, Equation (2). For that model the state space is defined as $s = (b_1, f_1, b_2, f_2) \in \mathcal{S}$. The resulting policy R is a four dimensional policy in $\mathcal{S}$:

$$R\,(s) = R\,(b_1, f_1, b_2, f_2)\,. \tag{48}$$

The simulation program uses the decisions given in $R\,(s)$ and keeps track of the information states and the actual number of foreground and background flows. In Figure 25 a sequence of information states for a node is plotted. In each bargraph title n is the observed and actual number of flows, f is the actual number of foreground flows and t is the time. On the horizontal axis the values correspond to the number of foreground flows. The height corresponds to the information state probability. In the first bargraph $t = 1$ the bar for value 2 is the highest so the system has most likely 2 foreground flows.

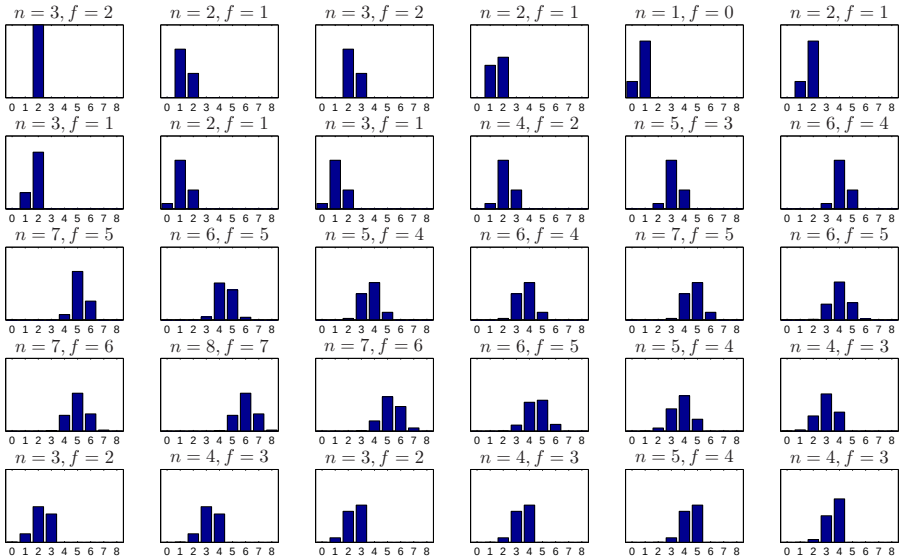


Figure 25: sequence of information states during simulation. The horizontal axis corresponds to the number of foreground flows, bar height corresponds to the probability for the corresponding number of flows on the horizontal axis.

As can be observed in the distribution plots, all distributions have a nice peaked curve shape. This observation generated the idea of fitting hypergeometric distributions to the information states. Finding a closed form for these fitted information state updates still is a difficult problem.

3.5 Combination of MDP policy with Bayesian information states

Another option is combining the (on line calculated) information state distribution information with the (off line calculated) solution of the fully observable MDP described in Equation (2) in Section 2.2. Let $\alpha(o_1, o_2)$ be the server selection decision based on the observed number of flows (o_1, o_2). Let $R(b_1, f_1, b_2, f_2)$ be the optimal policy for the fully observable MDP. For each of the two nodes the information state distribution is tracked. From these information states a mapping has to be made to server selection using the four dimensional policy from the full observable MDP. For the mapping two variants will be discussed:

Weighted in this variant decisions for all possible states, corresponding to the observed number of flows, are weighted using the information state probabilities. The result is rounded to the closest decision. For the weighted version define function $h(a)$ for $a \in \mathcal{A}$:

$$h(a) = \begin{cases} -1 \text{ for } a = 1 \\ 1 \text{ for } a = 2. \end{cases} \tag{49}$$

This will map the decision for server 1 to a negative value and the decision for server 2 to a positive value. Given the observed number of flows o_1 and o_2 and information state u the weighted variant will be:

$$\begin{aligned} \alpha(o_1, o_2) &= \sum_{i=0}^{o_1} \sum_{j=0}^{o_2} \left[h\left(R\left(o_1 - i, i, o_2 - j, j \right) u_{1o_1}(i) \, u_{2o_2}(j) \right) \right], \quad (50) \\ a &= \begin{cases} \text{server 1} & \text{if} \quad \alpha(o_1, o_2) < 0 \\ \text{server 2} & \text{if} \quad \alpha(o_1, o_2) > 0 \\ \text{server 1 or 2} & \text{otherwise.} \end{cases} \end{aligned}$$

Maximum likelyhood in this variant the most likely state is selected from the information state distribution. From the optimal policy $R(s)$ the action corresponding to the most likely states is chosen. In the maximum likelihood variant the most likely state is selected from the information states and plugged in the full observable MDP solution.

$$\begin{aligned} i &= \underset{i}{\operatorname{argmax}} \, u_{1o_1}(i), \\ j &= \underset{j}{\operatorname{argmax}} \, u_{2o_2}(j), \\ a &= \begin{cases} \text{server 1} & \text{if} \quad R\left(o_1 - i, i, o_2 - j, j \right) < 0 \\ \text{server 2} & \text{if} \quad R\left(o_1 - i, i, o_2 - j, j \right) > 0 \\ \text{server 1 or 2} & \text{otherwise.} \end{cases} \quad (51) \end{aligned}$$

3.6 Results and conclusion

Both the weighted combination and maximum likelyhood variants for combining information states with the solution of the full observeble MDP have been formulated and implemented in a realistic simulation environment. More details about the simulation can be found in Chapter 5. The simulation indicated that the weighted combination of Bayesian information states with the MDP did perform considerably worse than the maximum likelihood combination. From the simulation also the fraction of routed foreground files to each node has been stored (see Tables 15, 16, 17 in Chapter 5). In these fractions can be observed that for the weighted variant the fraction is always close to 50% while there this fraction should go away from 50% when the nodes are asymmetrically loaded or have different capacities. Somehow the weighting averages out the optimal decisions of the full observable MDP solution resulting in poor decisions.

4 Dynamic splitting heuristic based on conditional sojourn time

Consider the MDPs described in Chapter 2. In this case there are two nodes where on each node the total number of flows, o_1 and o_2, can be observed. The MDPs produce an optimal policy $R(o_1, o_2)$, on the observed number of flows, which optimizes the expected number of foreground flows. The policy provides server selection decisions that will route a new foreground file to a node based on the observed number of flows o_1 and o_2. In the MDP policies in Chapter 2 a nice structure can be observed (in Section 2.7). In this chapter a heuristic will be described that approximate the switching curve for systems with Poisson arrivals and exponential filesizes. For the optimization of expected sojourn time intuitively the optimal decisions should be based on the expected sojourn time for a new flow, conditioned on the observed number of flows. This chapter will start with a description of a method that enables the numerical calculation of the expected soujourn time given the observed number of flows and dynamic decision strategy (For example from a MDP). This will start with the analysed of a single PS node. In Section 4.1 the PS Markov chain is transformed in a uniformized Markov chain. The uniformized Markov chain is used to generate a probability tree conditioned on the inital observed state. Section 4.2 will present an algorithm that can calculate the expected sojourn time in a PS node for a new flow given that on arrival a specific number of flows is observed on that node. Section 4.3 will expand the single PS algorithm of section 4.2 to a multi-PS system including a policy on the observed number of flows. In Section 4.4 a server selection heuristic is defined, based on the knowledge aquired on conditional sojourn time in Section 4.2. Finally in Section 4.5 the performance, in terms of expected sojourn time, is compared for the heuristic in Section 4.4, join the shortest queue (JSQ) and the MDP (with all foreground and background flows observable) defined in equation (2) in Section 2.2.

In the MDP models the conditional expected sojourn time can not directly be derived because the MDPs states correspond with the number of flows on the PS-nodes. There is no exact expression for transforming the MDP states into conditional expected sojourn time, given a new transfer starts in state s, for a given dynamic policy. For the total expected sojourn time Little's formula can be applied using the expected number of flows in the system. The conditional sojourn time expectation however, can be derived numerically by using the properties of the continuous-time Markov chain. For now the problem will be simplified to a single PS-node.

4.1 Single PS conditional distribution

Consider the PS (processor sharing) continuous time Markov Chain with Poisson arrivals and exponential service times. This Markov chain has state space $\mathcal{S} = \mathbb{N}_0$ and $n, n' \in \mathcal{S}$ where n and n' represent the number of flows in the system. Further let $\lambda'(n, n')$ be the transition rate from state n to state n'. Figure 26 illustrates the well known PS Markov chain:

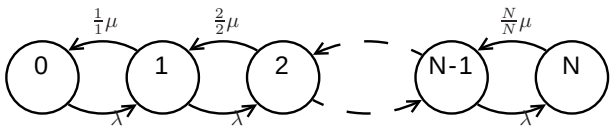


Figure 26: Processor sharing Markov chain.

The Markov chain has transition rates $\lambda'(n, n+1) = \lambda$ and $\lambda'(n, n-1) = \mu$.
This Markov chain can be uniformized by choosing a uniformization rate that is at least equal to the maximum of the sum of the rates $\lambda'(n, n')$ out of each state n:

$$\gamma \geq \max_n \left[\sum_n \lambda'(n, n') \right], \tag{52}$$

see uniformization in Section 8.A.3. For each state a dummy transition is added $\lambda'(n, n) = \gamma - \left[\sum_{n \neq n'} \lambda'(n, n') \right]$ such that for all states the sum of the rates out will be equal. The transition rates of the Markov chain can be transformed into transition probabilities due to the time independence property of the exponential distribution,

$$p(n, n') = \frac{\lambda'(n, n')}{\gamma}. \tag{53}$$

Now for every transition the expected transition time will be equal for each state. The expected transition time will be denoted by an epoch with expected length $\tau = \frac{1}{\gamma}$. Suppose the system is starting in a given state n. Then with the uniformized Markov chain the probability the system will be in state n' after t transition epochs can be calculated as follows:

- Let e_n be the unit vector with $e_n(n') = \mathbb{1}[n = n']$,
- let ν_0 be the initial probability distribution or knowledge on the number of flows,
- let $P =$ be the transition matrix with all uniformized transition probabilities.

For now the intitial knowledge ν_0 is chosen $\nu_0 = e_n$, so the distribution starts at the point where the number of flows is exactly known. Let $\mathbb{P}(n'|\nu_0, t)$ be the probability of observing the system in state n' after t uniformized transitions, with expected time $\frac{1}{\gamma}$, starting with knowledge ν_0. With $\nu_0 = e_n$ this is the probability that the system is found with n' flows while the system was observed for t epochs, starting with n flows at $t = 0$. This conditional probability can be calculated by

$$\mathbb{P}(n'|\nu_0, t) = e_{n'} P^t \nu_0. \tag{54}$$

The expected time for t epochs is equal to $t\tau = \frac{t}{\gamma}$.
Given the uniformized probabilities, a transition probability tree can be constructed, where the depth corresponds to the number of epochs:

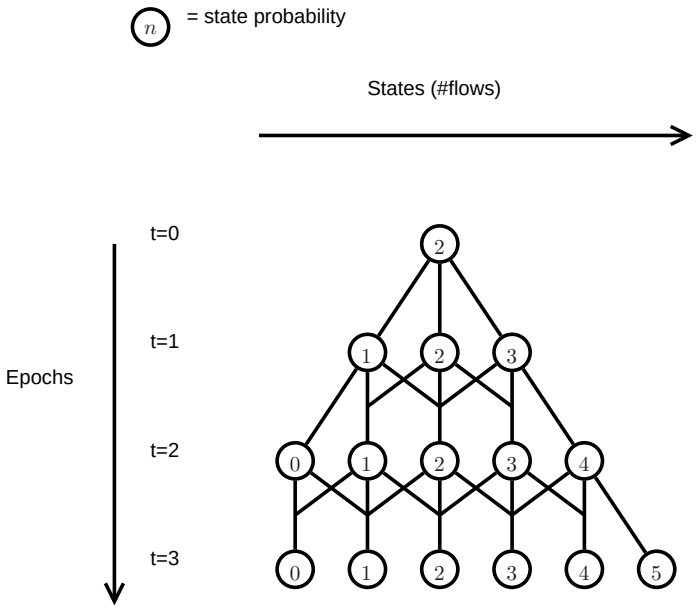


Figure 27: Probabilities after t epochs.

In Figure 27 the nodes correspond to states and the arcs correspond to the spreading of probability mass.

4.2 Conditional sojourn time algorithm for one PS node

Using the conditional probability in Section 4.1, the distribution on the number of flows after t epochs starting with ν_0 is known. This conditional distribution can be modified in order to be able to calculate the expected sojourn time for the given initial knowledge ν_0. First note that in a PS node, when there are k streams, each flow will obtain a fraction $\frac{1}{k}$ of the node capacity. When considering the uniformized Markov chain with departure transition rates μ each flow will be completed with probability $\frac{\mu}{k\gamma}$. Suppose a new flow arrives at $t = 0$ with n flows in the system. Then the new flow will be obtain a fraction $\frac{1}{n+1}$ of the capacity. Given the uniformized Markov chain the probability that after a transition this transition will be a flow completion is equal to $\frac{\lambda'(n,n-1)}{\gamma} = \frac{\mu}{\gamma} = \frac{\mu}{\gamma}$. Each flow has an equal probability of completion due to the independence property of the exponential distribution so the transition probability that a specific flow will be completed, with a new arrived flow, is equal to $\frac{\mu}{\gamma(n+1)}$. With this transition probability, conditioned on the new arrived flow, a sojourn time distribution $q(t)$ can be derived. Here $q(t)$ the distribution on the number

of epochs t before the newly added flow (on $t = 0$) will be completed. To the original Markov chain an extra state q' will be added. This state represents the situation where the new flow is completed.

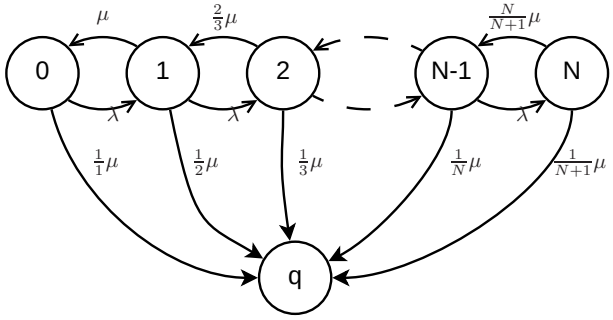


Figure 28: Modified PS (processor sharing) Markov Chain.

For the new flow, a state q' will be added to $\mathcal{S}$ resulting in state space $\mathcal{S}' = \{q', \mathbb{N}_0\}$. In this Markov chain the departure transition probabilities (n, n') are modified to $p'(n, n')$ where:

$$\begin{array}{lcll} \gamma & = & \lambda + \mu & \text{is the uniformization constant,} \\ p'\left(n, q'\right) & = & \dfrac{\mu}{\gamma(n+1)} & \text{is the probability that the new flow will be completed,} \\ p'\left(n, [n-1]^+\right) & = & \dfrac{n\mu}{\gamma(n+1)} & \text{is the probability that another flow will be completed,} \\ p'\left(n, n+1\right) & = & \dfrac{\lambda}{\gamma} & \text{is the probability that a new flow arrives.} \end{array} \tag{55}$$

Note that if the system is observed in state n the newly added flow will transfer the system in state $n+1$. Thus n corresponds to the number of flows additional to the flow added at $t = 0$. In state $n = 0$ no concurrent flows for the last added flow are in transfer. The new transition matrix will be denoted by P'. For the modified calculation define:

- e_n the unit vector with $e_n(n') = \mathbb{1}[n = n']$,
- ν_0 the initial probability distribution or knowledge on the number of flows,
- $P' =$ the modified transition matrix with all uniformized transition probabilities.

For the new transition matrix P' the conditional distribution given initial knowledge ν_0 and epoch t is defined by:

$$\mathbb{P}(n'|\nu_0, t) = e_{n'} {P'}^t \nu_0. \tag{56}$$

In this situation a probability tree can be constructed again where $q(t)$ represents the calculation for the departure probability of the new flow after t epochs. An example of such a tree is illustrated in Figure 29.

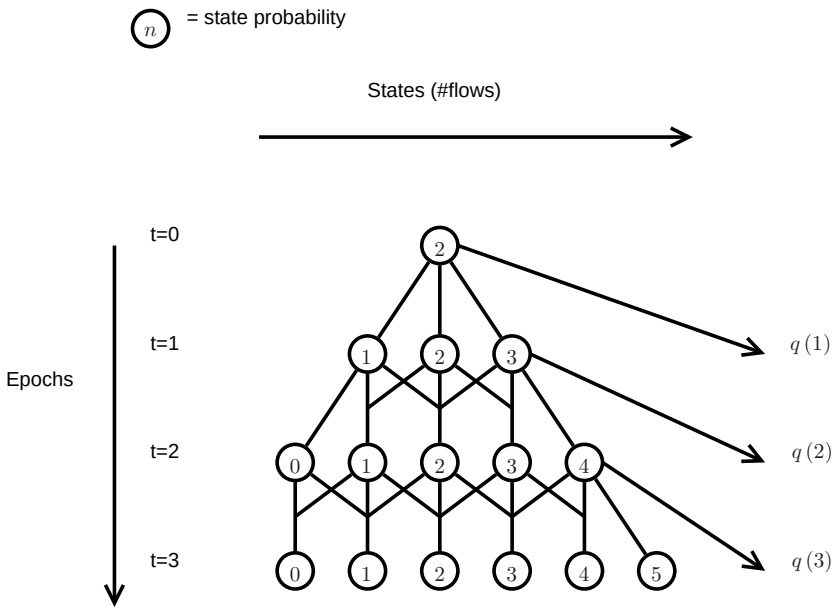


Figure 29: $q(t)$ represents the calculation for the departure probability of the new flow after t epochs.

The cumulative sojourn time distribution $Q(t)$ is embedded in the sequence of conditional distributions $\mathbb{P}(n'|\nu_0, t)$ for $t = 0, \cdots, T$. where T goes to infinity:

$$Q(t) = \mathbb{P}(q'|\nu_0, t) = e_{n'} P'^t \nu_0. \tag{57}$$

For each epoch $q(t)$ the departure probability of the last added flow $p'(n, q')$ can be calculated by summing up the probabilities that the system has n flows at time t and the next event is a departure of the flow added at $t = 0$:

$$q(t) = \sum_n \left[\mathbb{P}(q'|\nu_0, t-1) \cdot p'(n, q')\right] = \sum_n \left[\left(e_{n'} P'^t \nu_0\right) \frac{\mu}{\gamma(n+1)}\right]. \tag{58}$$

As the distribution of q represents the number of epochs before the new flow is completed and the expected length of an epoch is known, the conditional expected sojourn time $\mathbb{E}\left[S\middle|\nu_0\right]$ can be expressed in terms of q:

$$\mathbb{E}\left[S\middle|\nu_0\right] = \frac{1}{\gamma} \sum_{t=0}^{T} \left[t q(t)\right]. \tag{59}$$

Typically the sojourn time can be conditioned on the number of flows n with $S_n = \mathbb{E}\left[S|n\right]$. Therefore for each starting state the starting distribution is defined by:

$$\nu_{0,n} = e_n. \tag{60}$$

The calculation of the contitional sojourn time can be transformed in an algorithm:

$S_n \leftarrow 0$
$t \leftarrow 0$
$\nu_{0,n} \leftarrow e_n$
$p\left(n'|n,t\right) \leftarrow \nu_{0,n}\left(n'\right)$
transition matrix P'
while diff $> \epsilon$ **do**
$q\left(t,n\right) \leftarrow \sum_{n'} p\left(n'|n,t\right) \frac{\mu}{\left(n'+1\right)\gamma}$
$S_n \leftarrow S_n + \frac{t}{\gamma} q\left(t,n\right)$
diff $\leftarrow \max_n |S_n - S'_n|$
$p\left(n'|n,t\right) \leftarrow p\left(n'|n,t-1\right) P'$
$S'_n \leftarrow S_n$
end while

The algorithm will stop when the desired accuracy ϵ is reached.

4.2.1 Conditional sojourn time a on single PS node

When the algorithm given in last paragraph was implemented and the results were evaluated, a linear relation was found between conditioned state n and expected conditional sojourn time S_n

$$S_n = \frac{n+2}{2\mu - \lambda}. \tag{61}$$

This result can also be found in [9], which describes a polynomial relationship between the n'th moment of the conditioned sojourn time and state n. The first moment exactly corresponds to the conditional expected sojourn time found using the algorithm.

4.3 Multi PS node algorithm with server selection policy

For a single PS node without server selection policy the calculation is trivial. However when the calculation is extended to the multi-PS node case with a dynamic server selection policy the exact expression for the conditional expected sojourn time becomes hard to calculate. Consider the model in Equation (2), Section 2.2. Now the policy determines the arcs for foreground traffic between the different states. In the example below there are two nodes where on foreground arrival one node has to be selected.

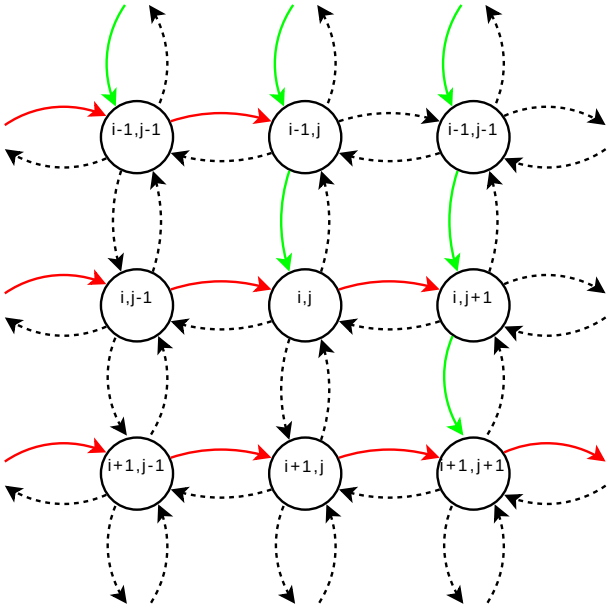


Figure 30: Server selection Markov chain.

The black arrows are regular transitions, the red arrows are the case server 2 is selected, the green arrows are the case server 1 is selected on a new foreground arrival.

Now a state space $\mathcal{S} = {\mathbb{N}_0}^2$ is defined with $(n_1, n_2) \in \mathcal{S}$ where n_1 corresponds to the number of flows on server 1 and n_2 corresponds to the number of flows on server 2. The modified statepace will become $\mathcal{S}' = \{q'\} \cup \mathcal{S}$ with q' the flow completion state. States will be denoted as tuples on the number of flows $[n_1, n_2]$ and $[q']$ for the exit state. From this Markov chain two modified transition matrices P_1' and P_2' can be constructed. To this transition matrix the dynamic policy $R\,(n_1, n_2)$ with server selection decisions for given n_1 and n_2 and selected server $a \in \{1, 2\}$ has to be provided:

$$
\begin{aligned}
\gamma &= \lambda_0 + \lambda_1 + \lambda_2 + \mu_1 + \mu_2, \\
P_1' &:= \\
p_1'\Big(\big[n_1, n_2\big], \big[q'\big]\Big) &= \frac{\mu_1}{\gamma\left(n_1+1\right)}, \\
p_1'\Big(\big[n_1, n_2\big], \big[[n_1-1]^+, n_2\big]\Big) &= \frac{n_1\mu_1}{\gamma\left(n_1+1\right)}, \\
p_1'\Big(\big[n_1, n_2\big], \big[n_1+1, n_2\big]\Big) &= \frac{\lambda_1 + \mathbb{1}\left[R\left(n_1, n_2\right)=1\right]\lambda_0}{\gamma}, \\
p_1'\Big(\big[n_1, n_2\big], \big[n_1, [n_2-1]^+\big]\Big) &= \frac{\mu_2}{\gamma}, \\
p_1'\Big(\big[n_1, n_2\big], \big[n_1, n_2+1\big]\Big) &= \frac{\lambda_2 + \mathbb{1}\left[R\left(n_1, n_2\right)=2\right]\lambda_0}{\gamma}, \\
P_2' &:= \\
p_2'\Big(\big[n_1, n_2\big], \big[[n_1-1]^+, n_2\big]\Big) &= \frac{\mu_1}{\gamma}, \\
p_2'\Big(\big[n_1, n_2\big], \big[n_1+1, n_2\big]\Big) &= \frac{\lambda_1 + \mathbb{1}\left[R\left(n_1, n_2\right)=1\right]\lambda_0}{\gamma}, \\
p_2'\Big(\big[n_1, n_2\big], \big[q'\big]\Big) &= \frac{\mu_2}{\gamma\left(n_2+1\right)}, \\
p_2'\Big(\big[n_1, n_2\big], \big[n_1, [n_2-1]^+\big]\Big) &= \frac{n_2\mu_2}{\gamma\left(n_2+1\right)}, \\
p_2'\Big(\big[n_1, n_2\big], \big[n_1, n_2+1\big]\Big) &= \frac{\lambda_2 + \mathbb{1}\left[R\left(n_1, n_2\right)=2\right]\lambda_0}{\gamma}.
\end{aligned}
$$

From this the sojourn time distribution for both servers can be expressed. Define:

- e_{n_1,n_2} the unit vector with $e_{n_1,n_2}\left(n_1', n_2'\right) = \mathbb{1}\left[n_1 = n_1' \wedge n_2 = n_2'\right]$,
- ν_0 the initial probability distribution or knowledge on the number of flows,
- P_1' and P_2' the modified transition matrix with all uniformized transition probabilities for node 1 and 2,
- $\mathbb{P}_a\left(\big[q'\big]\Big|\nu_0, t\right) = e_{n_1',n_2'}{P_a'}^t\nu_0$ the probability distribution on the number of flows given intial knowledge ν_0 and epoch t for node a.

For each epoch $q_a\left(t\right)$ the departure probability of the last added flow $p_a'\left(n_a, q_a'\right)$ can be calculated by summing up the probabilities system a has n_a flows at time t and the next event is a departure of the flow added at $t=0$:

$$
\begin{aligned}
q_1\left(t\right) &= \sum_{n_1,n_2}\left[\mathbb{P}_1\left(\big[q'\big]\Big|\nu_0, t-1\right)\cdot p_1'\left(\big[n_1,n_2\big],\big[q'\big]\right)\right] = \sum_{n_1,n_2}\left[\left(e_{n_1',n_2'}{P_1'}^t\nu_0\right)\frac{\mu_1}{\gamma\left(n_1+1\right)}\right], \\
q_2\left(t\right) &= \sum_{n_1,n_2}\left[\mathbb{P}_2\left(\big[q'\big]\Big|\nu_0, t-1\right)\cdot p_2'\left(\big[n_1,n_2\big],\big[q'\big]\right)\right] = \sum_{n_1,n_2}\left[\left(e_{n_1',n_2'}{P_2'}^t\nu_0\right)\frac{\mu_2}{\gamma\left(n_2+1\right)}\right].
\end{aligned}
$$

As the distribution of q_a represents the number of epochs before the new flow is completed on node $a=1,2$ and the expected length of an epoch is known, the conditional expected sojourn time $\mathbb{E}\left[S_a\Big|\nu_0\right]$ can be expressed in terms of q_a:

$$
\mathbb{E}\left[S_a\Big|\nu_0\right] = \frac{1}{\gamma}\sum_{t=0}^{T}\left[tq_a\left(t\right)\right]. \tag{62}
$$

The expected sojourn time can be conditioned on the number of flows n_1, n_2 on both servers for selected server $a \in \{1, 2\}$ with $S_{a,n_1,n_2} = \mathbb{E}\left[S_a \middle| n_1, n_2\right]$. Therefore for each starting state the starting distribution is defined by:

$$\nu_{0,n_1,n_2} = e_{n_1,n_2}. \tag{63}$$

Now the algorithm will be:

$S_{a,n_1,n_2} \leftarrow 0$
$t \leftarrow 0$
$\nu_{0,n_1,n_2} \leftarrow e_{n_1,n_2}$
$p_a\left(n_1', n_2' \middle| n_1, n_2, t\right) \leftarrow \nu_{0,n_1,n_2}\left(n_1', n_2'\right)$
transition matrices P_1', P_2'
while diff $> \epsilon$ **do**
  $q_a\left(t, n_1, n_2\right) \leftarrow \sum_{n_1', n_2'} p_a\left(n_1', n_2' | n_1, n_2, t\right) \frac{\mu_a}{\left(n_a' + 1\right)\gamma}$
  $S_{a,n_1,n_2} \leftarrow S_{a,n_1,n_2} + \frac{t}{\gamma} q_a\left(t, n_1, n_2\right)$
  diff $\leftarrow \max_{n_1,n_2} \left|S_{a,n_1,n_2} - S'_{a,n_1,n_2}\right|$
  $p_a\left(n_1', n_2' | n_1, n_2, t\right) \leftarrow p_a\left(n_1', n_2' | n_1, n_2, t-1\right) P_a'$
  $S'_{a,n_1,n_2} \leftarrow S_{a,n_1,n_2}$
end while

The algorithm will produce for each server $a = 1, 2$ a matrix S_{a,n_1,n_2} on the (n_1, n_2) plane with the expected sojourn time conditioned on (n_1, n_2). These two matrices can be combined in one conditional sojourn time expression using the dynamic server selection policy $R\left(n_1, n_2\right)$ that is used in the calculation:

$$\mathbb{E}\big[S\big|n_1, n_2\big] = \mathbb{1}\big[R\left(n_1, n_2\right) = 1\big] S_{1,n_1,n_2} + \mathbb{1}\big[R\left(n_1, n_2\right) = 2\big] S_{2,n_1,n_2}. \tag{64}$$

4.4 Optimization heuristic based on conditioned sojourn time

For the algorithm in Section 4.3 a simple server selection policy can be provided for testing. The policy that is used is the join the shortest queue policy. This policy will be evaluated in a realistic simulation environment in Chapter 5. Figure 31 represents the conditional expected sojourn time for the join the shortes queue policy:

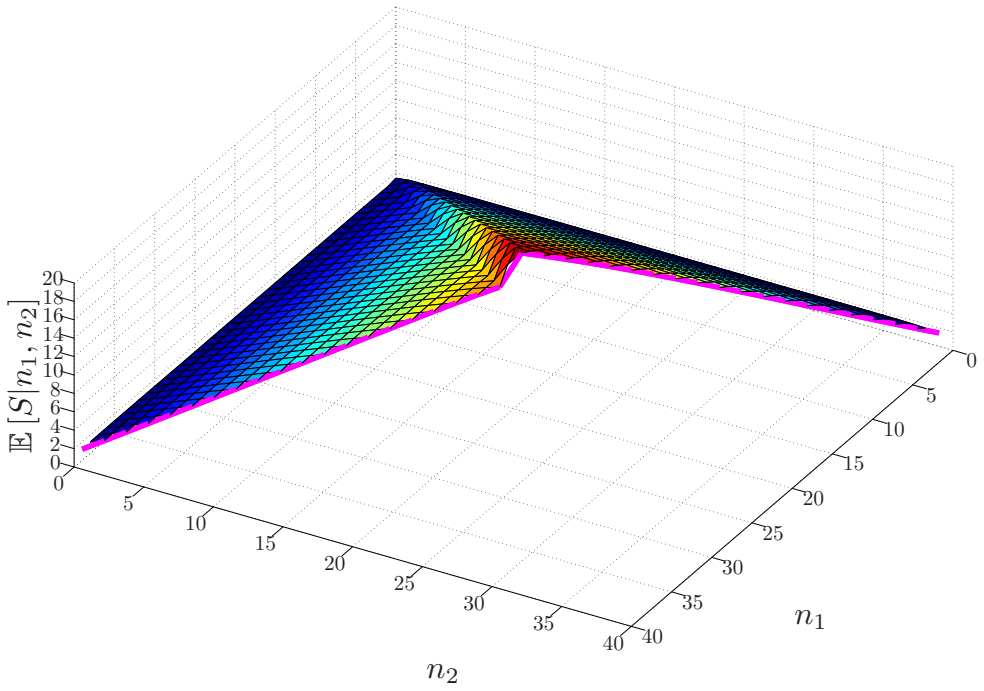


Figure 31: Back view.

This graph has parameters:

- $\lambda_0 = .1$
- $\lambda_1 = .3$
- $\lambda_2 = .7$
- $\mu_1 = 1$
- $\mu_2 = 1$

In Figure 31 a pyramid shape can be recognized with a gap over the diagonal. This gap can be observed in the purple line which corresponds to the expected sojourn time over the line $n_1 + n_2 = 40$. When the three-dimensional graph of the conditional sojourn time is observed from another perspective in Figure 32, a kind of pyramid can be recognized with a gap in the intersection of sides.

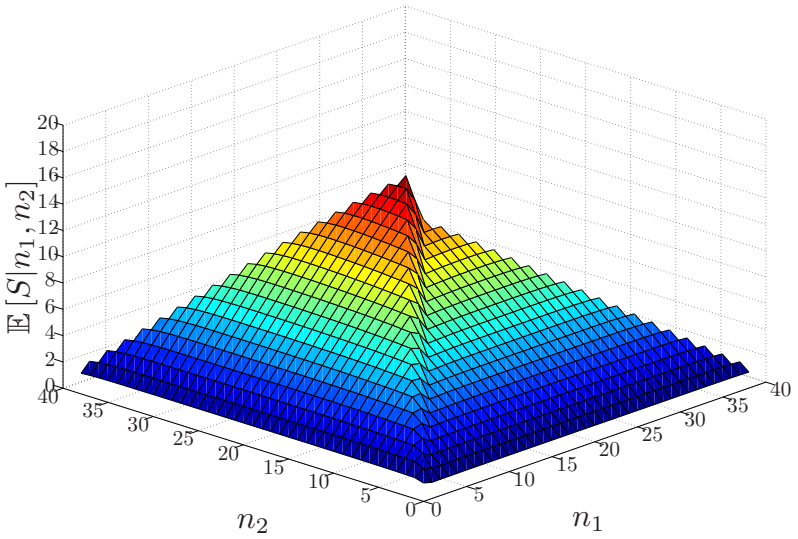


Figure 32: Front view.

Consider the virtual sides of the almost pyramid shape in Figure 32. The expected sojourn time, in this graph, can be observed as the minimum of two almost flat planes in the n_1, n_2 surface. If the decision switching curve was chosen different the gap can be removed. This brought the idea that foreground sojourn time can be optimized by choosing the switching curve on the intersection of the conditional sojourn time planes for both servers. For relatively low foreground traffic the conditional sojourn time planes are close to the actual planes. Therefore the intersection of those planes can be simplified by calculating the intersection of the simplified conditional sojourn time planes. Consider the expression of conditional sojourn time for a single-PS node in Section 4.2.1. Given the background flows arrival rates for nodes 1 and 2, λ_1, λ_2 and the capacities for both nodes μ_1 and μ_2 for both nodes the conditional sojourn time can be expressed in the (n_1, n_2) plane using the single-PS conditional sojourn time expression. For node $a = 1, 2$ the plane will be described by:

$$S_a(n_1, n_2) = \frac{n_a + 2}{2\mu_a - \lambda_a},$$

where $S_a(n_1, n_2)$ is the expression for the sojourn time on node a conditioned on (n_1, n_2). The switching curve can now be defined by the solution of the equation:

$$S_1(n_1, n_2) = S_2(n_1, n_2), \tag{65}$$

resulting in equation:

$$\frac{n_1 + 2}{2\mu_1 - \lambda_1} = \frac{n_2 + 2}{2\mu_2 - \lambda_2}. \tag{66}$$

Figure 33 illustrates the idea of intersecting conditional sojourn time planes $S_1(n_1, n_2)$ and $S_2(n_1, n_2)$.

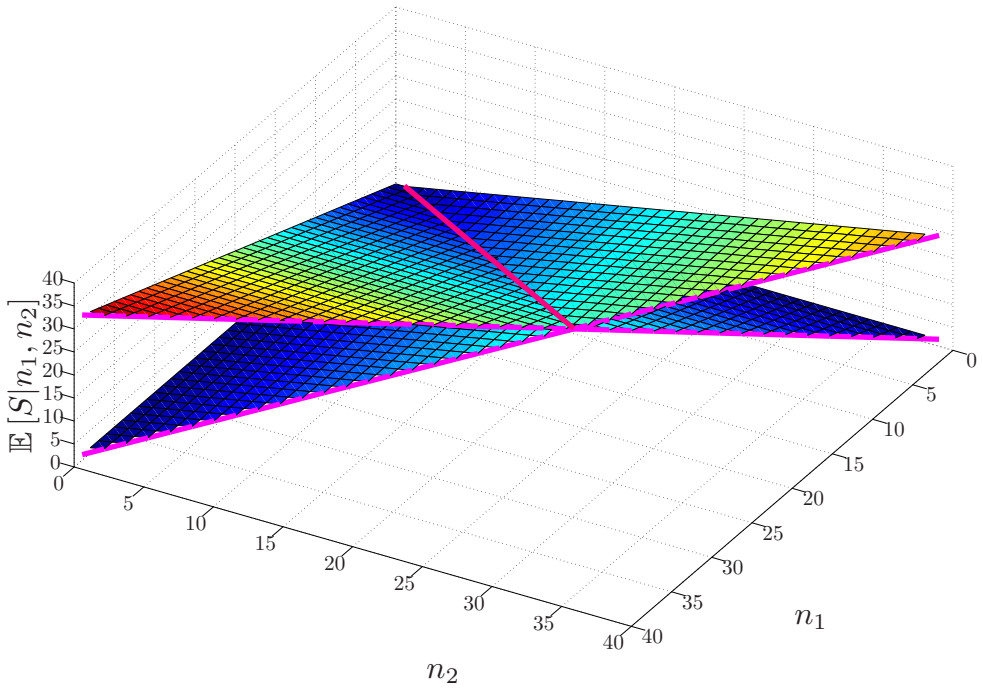


Figure 33: Intersecting $\mathbb{E}[S]$ planes.

Multidimensional case The conditional sojourn time server selection regime can easily be extended to the case with more than two servers. Let $a \in \mathcal{A}$ be the set of selectable servers. In that case the optimal selection is defined by:

$$\min_a \left\{ \frac{n_a + 2}{2\mu_a - \lambda_a} \right\}. \tag{67}$$

with

n_a the number of flows on server a,

λ_a the background file arrival rate on server a,

μ_a the capacity of server a.

4.5 Performance evaluation of the conditional sojourn time heuristic

It would be interesting to know how the server selection policy based on conditional sojourn time performs compared to other policies. Therefore the expected

sojourn time is calculated and compared for three different policies

- $\mathbb{E}[S|CondES]$ conditional sojourn time policy,
- $\mathbb{E}[S|JSQ]$ join the shortest queue,
- $\mathbb{E}[S|MDP]$ full MDP is Equation (2), Section 2.2.

As described in section 2.2 we consider a system with two nodes with on each node a number of foreground flows f_a and background flows b_a for $a = 1, 2$. The idea is to optimize the expected sojourn time for foreground flows for a given policy. The policies from $\mathbb{E}[S|CondES]$ and $\mathbb{E}[S|JSQ]$ are defined on the total number of flows observed on each server $o_a = f_a + b_a$. Using the expression for the total number of flows the full observable model can be used with these policies. For these policies the Markov chain was solved for different parameters. For the three policies a continuous time Markov chain can be solved that incorporates the decisions on foreground flow arrivals. From the Markov chain solution the expected number of foreground flows was determined and using Little's formula the expected sojourn time was calculated. For the comparison the foreground arrival rate and the file sizes have been chosen fixed:

$\lambda_0 = .1,$

$\mu_0 = \mu_1 = \mu_2 = 1.$

The parameter value $\lambda_0 = .1$ represents the assumption that the amount foreground traffic is small compared to the background traffic. The file size parameters are equal for all flows $\mu_0 = \mu_1 = \mu_2 = 1$ for sake of simplicity and because the impact of difference in server capacity will be evaluated. For varied background arrival rates λ_1 and λ_2 the expected sojourn time was calculated for the three policies. As the full MDP gives the optimal policy the expected sojourn time of this policy is used as performance benchmark. The tables contain the relative difference in expected sojourn time to the full MDP expected sojourn time. This percentage is calculated by:

$$\frac{\mathbb{E}[S|\text{test policy}] - \mathbb{E}[S|MDP]}{\mathbb{E}[S|MDP]} \times 100\%. \tag{68}$$

First the percentages were calculated for equal capacity for both nodes:

Table 4: Equal capacity $C_1 = C_2 = 1$, full MDP v.s. CondES.

λ_1/λ_2	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	0.26%	0.51%	0.56%	0.49%	0.35%	0.39%	0.24%	0.86%	0.34%
0.2		0.42%	0.62%	0.59%	0.46%	0.34%	0.40%	1.14%	0.37%
0.3			0.52%	0.66%	0.59%	0.45%	0.34%	0.37%	0.53%
0.4				0.57%	0.66%	0.55%	0.43%	0.64%	0.26%
0.5					0.58%	0.63%	0.51%	0.54%	0.41%
0.6						0.55%	0.56%	0.50%	0.43%
0.7							0.50%	0.48%	0.50%
0.8								0.42%	0.47%
0.9									0.38%

Table 5: Equal capacity $C_1 = C_2 = 1$, full MDP v.s. JSQ.

λ_1/λ_2	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	0.26%	1.89%	3.21%	4.19%	4.79%	4.99%	4.90%	4.24%	2.86%
0.2		0.42%	1.77%	2.82%	3.53%	3.87%	3.86%	3.52%	2.45%
0.3			0.52%	1.62%	2.42%	2.89%	3.01%	2.80%	2.06%
0.4				0.57%	1.44%	2.02%	2.25%	2.17%	1.67%
0.5					0.58%	1.24%	1.59%	1.61%	1.29%
0.6						0.55%	1.01%	1.14%	0.93%
0.7							0.50%	0.75%	0.64%
0.8								0.42%	0.47%
0.9									0.38%

In Tables 4 and 5 it can be observed that the maximal difference to the MDP for the conditional sojourn time is 1.14% and for JSQ 4.99%. Only the upper triangle of the tables is displayed here, because all differences are symmetrical due to the equal server capacity $C_1 = C_2$. Tables 6 and 7 contain the relative difference with unequal ratio of server capacities starting with $C_1 = 1, C_2 = 2$:

Table 6: Unequal capacity $C_1 = 1, C_2 = 2$, full MDP v.s. CondES.

λ_1/λ_2	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	0.37%	0.28%	0.19%	0.12%	0.08%	0.05%	0.02%	0.01%	0.00%
0.2	0.46%	0.42%	0.29%	0.18%	0.15%	0.07%	0.03%	0.01%	0.00%
0.3	0.32%	0.57%	0.40%	0.25%	0.12%	0.11%	0.06%	0.03%	0.01%
0.4	0.44%	0.54%	0.51%	0.33%	0.18%	0.19%	0.09%	0.06%	0.01%
0.5	0.53%	0.27%	0.65%	0.42%	0.25%	0.10%	0.13%	0.05%	0.02%
0.6	0.61%	0.37%	0.58%	0.51%	0.32%	0.16%	0.18%	0.08%	0.05%
0.7	0.66%	0.45%	0.25%	0.66%	0.39%	0.24%	0.32%	0.11%	0.08%
0.8	0.70%	0.51%	0.34%	0.60%	0.47%	0.28%	0.13%	0.16%	0.05%
0.9	0.85%	0.57%	0.40%	0.25%	0.62%	0.34%	0.21%	0.25%	0.07%

Table 7: Unequal capacity $C_1 = 1, C_2 = 2$, full MDP v.s. JSQ.

λ_1/λ_2	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	39.53%	39.86%	39.58%	38.54%	36.56%	33.43%	28.75%	22.08%	12.99%
0.2	36.69%	37.26%	37.52%	37.04%	35.61%	32.99%	28.77%	22.40%	13.36%
0.3	33.67%	34.42%	35.20%	35.27%	34.40%	32.31%	28.61%	22.62%	13.70%
0.4	30.56%	31.68%	32.63%	33.25%	32.94%	31.41%	28.26%	22.72%	14.00%
0.5	27.41%	28.85%	29.91%	30.99%	31.25%	30.31%	27.71%	22.71%	14.26%
0.6	24.28%	25.98%	27.36%	28.54%	29.35%	28.99%	26.97%	22.55%	14.48%
0.7	21.21%	23.12%	24.76%	26.00%	27.25%	27.49%	26.10%	22.26%	14.65%
0.8	18.24%	20.28%	22.14%	23.68%	24.99%	25.80%	25.04%	21.82%	14.77%
0.9	15.41%	17.52%	19.53%	21.32%	22.69%	23.95%	23.84%	21.32%	14.84%

In these tables it can be observed that the difference between the expected conditional sojourn time policy and the MDP is smaller while this difference is increased between the JSQ and MDP policy. For unequal capacity the comparision with JSQ is not completely fair. Therefore also the JSQ variant is considered where the switching curve is determined by the ratio of foreground loads on both servers:

$$a = \begin{cases} \text{server 1} & \text{if} \quad n_1 < n_2 \dfrac{C_1}{C_2} \\ \text{server 2} & \text{if} \quad n_1 > n_2 \dfrac{C_1}{C_2} \\ \text{server 1 or 2} & \text{otherwise.} \end{cases} \tag{69}$$

The percentage difference in expected sojourn time for this policy to the MDP is:

Table 8: Unequal capacity $C_1 = 1, C_2 = 2$, full MDP v.s. JSQ adjusted to foreground load ratio.

λ_1/λ_2	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	38.60%	38.50%	37.80%	36.38%	34.10%	30.79%	26.14%	19.81%	11.50%
0.2	35.27%	35.20%	34.83%	33.77%	31.87%	28.97%	24.77%	18.91%	11.05%
0.3	31.85%	31.79%	31.74%	31.04%	29.53%	27.03%	23.32%	17.95%	10.58%
0.4	28.42%	28.58%	28.55%	28.21%	27.09%	25.01%	21.78%	16.93%	10.08%
0.5	25.02%	25.39%	25.33%	25.31%	24.58%	22.93%	20.14%	15.85%	9.54%
0.6	21.72%	22.26%	22.41%	22.35%	22.01%	20.77%	18.42%	14.69%	8.96%
0.7	18.56%	19.24%	19.57%	19.46%	19.40%	18.57%	16.69%	13.45%	8.35%
0.8	15.55%	16.35%	16.83%	16.91%	16.79%	16.35%	14.91%	12.16%	7.69%
0.9	12.75%	13.61%	14.21%	14.47%	14.29%	14.12%	13.11%	10.86%	6.98%

This is slightly better than the original JSQ but considerably worse than the conditional sojourn time based policy. When the difference of server capacity is increased to $C_1 = 1, C_2 = 4$ the percentage difference in expected sojourn time between the conditional sojourn time policy and MDP is very close to zero.

Table 9: Unequal capacity $C_1 = 1, C_2 = 4$, full MDP v.s. CondES.

λ_1/λ_2	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.2	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.3	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.4	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.5	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.6	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.7	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.8	0.01%	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.9	0.01%	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

The difference between JSQ and MDP increased to even larger percentages:

Table 10: Unequal capacity $C_1 = 1, C_2 = 4$, full MDP v.s. JSQ.

λ_1/λ_2	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	133.11%	128.02%	121.56%	113.43%	103.30%	90.69%	75.02%	55.49%	31.45%
0.2	131.72%	127.26%	121.37%	113.77%	104.08%	91.80%	76.28%	56.68%	32.29%
0.3	130.13%	126.29%	121.01%	113.96%	104.74%	92.82%	77.50%	57.87%	33.13%
0.4	128.33%	125.13%	120.46%	113.99%	105.27%	93.75%	78.68%	59.05%	33.98%
0.5	126.32%	123.76%	119.72%	113.85%	105.68%	94.59%	79.80%	60.21%	34.84%
0.6	124.10%	122.18%	118.79%	113.54%	105.94%	95.33%	80.86%	61.35%	35.72%
0.7	121.66%	120.39%	117.65%	113.04%	106.05%	95.96%	81.86%	62.48%	36.60%
0.8	119.02%	118.38%	116.30%	112.36%	106.00%	96.48%	82.79%	63.59%	37.49%
0.9	116.15%	116.15%	114.74%	111.48%	105.79%	96.87%	83.65%	64.67%	38.40%

The adjusted JSQ does not really perform better:

Table 11: Unequal capacity $C_1 = 1, C_2 = 4$, full MDP v.s. JSQ adjusted to foreground load ratio.

λ_1/λ_2	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	131.52%	125.68%	118.56%	109.90%	99.40%	86.67%	71.20%	52.29%	29.43%
0.2	129.25%	123.62%	116.71%	108.28%	98.02%	85.54%	70.32%	51.68%	29.11%
0.3	126.78%	121.37%	114.70%	106.51%	96.50%	84.28%	69.34%	51.00%	28.75%
0.4	124.11%	118.94%	112.51%	104.57%	94.83%	82.90%	68.26%	50.25%	28.36%
0.5	121.24%	116.32%	110.15%	102.48%	93.03%	81.40%	67.09%	49.44%	27.92%
0.6	118.19%	113.52%	107.62%	100.25%	91.10%	79.79%	65.83%	48.56%	27.45%
0.7	114.95%	110.55%	104.93%	97.86%	89.03%	78.07%	64.48%	47.61%	26.95%
0.8	111.53%	107.41%	102.09%	95.33%	86.84%	76.25%	63.05%	46.61%	26.42%
0.9	107.93%	104.10%	99.08%	92.66%	84.53%	74.31%	61.54%	45.55%	25.85%

This brings the question what is the difference between the adjusted JSQ and CondES? Now lets compare the policies of the corrected JSQ and CondES.

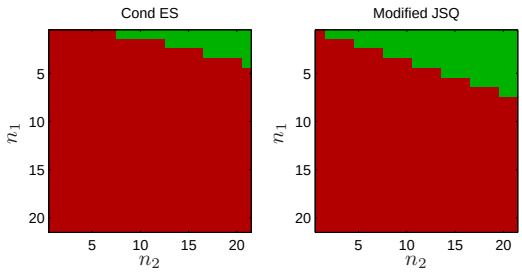


Figure 34: Comparison of policies.

Figure 34 shows that the biggest difference lies in the threshold of the CondES policy. This policy first sends all files to the node with highest capacity until a certain threshold is reached. This brings the conjecture that this threshold behavior is crucial for the performance of server selection policies. This can be explained by considering the distribution on the number of flows in the system. Most likely the system is found with a small number of flows. Therefore the impact of decisions in that area has a large contribution to the expected sojourn time.

4.6 Conclusion

In this chapter a server selection heuristic has been developed that is based on the expression of conditional sojourn time in a single-PS node. The performance of the heuristic has been compared to JSQ and the full observable MDP found in Equation (2), Section 2.2. The full observable MDP is considered as a benchmark case, because here all information on the number of flows is used for optimal decisions. The comparison been done for both symmetric and asymmetric node capacities. For symmetrical capacity the performance of the conditional sojourn time heuristic is close to the full MDP solution (in the order of 1%). When the asymmetry beween server capacity grows the difference in expected sojourn time of the conditional sojourn time server server selection and the fully observable MDP becomes really small. This means that using the conditional sojourn time strategy gives performance that is really close tho the MDP solution. Now it is not necessary anymore to solve the MDP for obtaining almost equal performance. It is suitable to apply the conditional sojourn time split which can be determined on line.

5 Simulation results in OPNET

In the previous chapters, different server selection strategies are proposed. For all the server selection strategies it would be interesting to examine the performance in terms of expected sojourn time in a more realistic setting. OPNET is a simulation environment that implements the full range of lower-layer protocol details [6]. We have implemented a scenario with two wireless LANs.

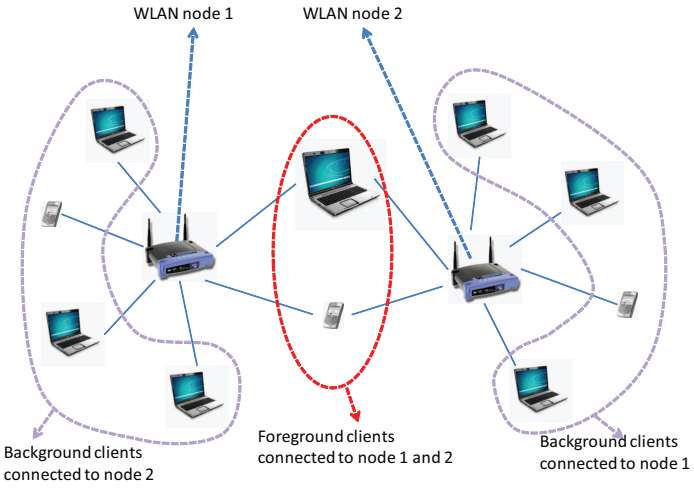


Figure 35: WLAN concurrent access simulation setup.

Each of the wireless LANs has clients that only use one access point. These clients represent the background traffic. There is also a group of clients that can use both access point simultaneously. These clients represent the foreground traffic. Both access points are modeled with the IEEE 802.11b standard with a maximal throughput of $11Mbit$. The clients generate requests using a Poisson distribution. The access points are connected to a FTP server that contains files which can be requested by the clients. Furthermore the filesize distributions are drawn from the exponential distribution. For the foreground clients multiple server selection strategies have been implemented in OPNET:

1. **PO MDP**, the partial observable MDP from Equation (11) in Section 2.4,
2. **Inf state ML**, the full observable MDP strategy from Section 2.2 combined with Bayesian information states using the maxmimum likelihood as described in Chapter 3,
3. **Inf state W**, the full observable MDP strategy from Equation (11) in Section 2.4 combined with Bayesian information states using weighting over the information state distribution as described in Chapter 3,
4. **Cond E[S]**, the conditional sojourn time strategy described in Section 4.4
5. **JSQ**, Join the Shortest Queue.

These policies have been simulated using different scenarios comprising both symmetric and assymmetric node capacities.

Table 12: Simulation scenarios.

	Bandwidth WLAN node 1	Bandwidth WLAN node 2
Scenario 1	$11Mbit/s$	$11Mbit/s$
Scenario 2	$11Mbit/s$	$5.5Mbit/s$
Scenario 3	$11Mbit/s$	$1Mbit/s$

For the different scenarios the WLAN transmission rates and file requests have to be translated in a set of parameters that can be used for calculating the different server selection strategies that will be tested. We have developed a model where the properties of the clients and the network can be translated in file arrival rates $\lambda_0, \lambda_1, \lambda_2$ and server capacities C_1, C_2 [5]. For the scenarios the following set of parameters was used:

Table 13: Simulation parameters.

	λ_0	λ_1	λ_2	C_1	C_2	$C_1 : C_2$
Scenario 1	0.298	2.087	0.894	2.982	2.982	1 : 1
Scenario 2	0.298	2.087	0.604	2.982	2.016	1 : 1.48
Scenario 3	0.298	2.087	0.154	2.982	0.515	1 : 1.579

. The effective loads in Table 14 can be obtained by plugging in $\lambda_0, \lambda_1, \lambda_2$ and C_1, C_2 from table 13 into the expression $\rho_{i,j} = \frac{\lambda_i}{C_j}$ where $i \in \{0, 1, 2\}$ and $j \in \{1, 2\}$.

Table 14: Simulation effective loads.

	ρ_0 node 1	ρ_0 node 2	ρ_1 node 1	ρ_2 node 2
Scenario 1	0.1	0.1	0.7	0.3
Scenario 2	0.1	0.148	0.7	0.3
Scenario 3	0.1	0.579	0.7	0.3

From the OPNET simulations each scenario did generate for the different server selection strategies a series containing the foreground sojourn times. From these sojourn times using the batch means procedure the means and the 95% confidence interval was determined. The results for the different server selection strategies are presented in Figures 36, 37 and 38.

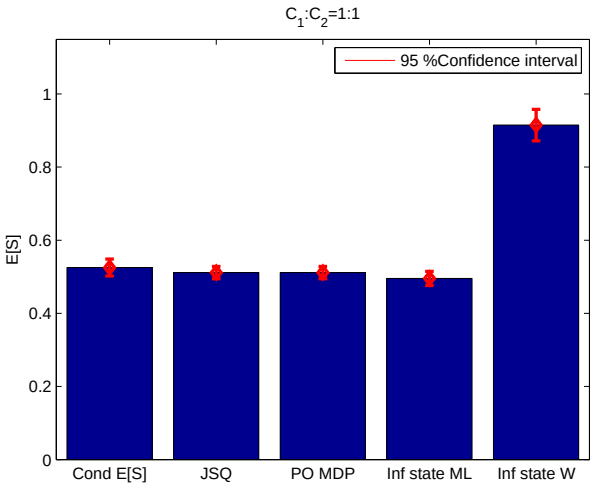


Figure 36: OPNET results for $C_1 : C_2 = 1 : 1$.

Table 15: Fraction of foreground files routed to node 1 for $C_1 : C_2 = 1 : 1$.

Cond E[S]	JSQ	PO MDP	Inf state ML	Inf state W
0.6301	0.8506	NA	0.8618	0.5106

In Figure 36 can be observed that the average sojourn time for foreground files is close for all strategies except the **Inf state W** policy. This policy performs consideraby worse.

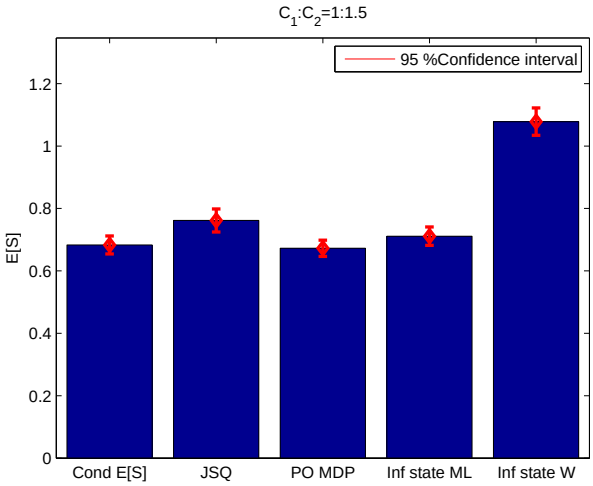


Figure 37: OPNET results for $C_1 : C_2 = 1 : 1.48$.

Table 16: Fraction of foreground files routed to node 1 for $C_1 : C_2 = 1 : 1.48$.

Cond E[S]	JSQ	PO MDP	Inf state ML	Inf state W
0.6078	0.8379	NA	0.6204	0.4828

In Figure 37 **Inf state W** policy still performs consideraby worse. For the **JSQ** the expected sojourn time is also higher. This difference lies outside the 95% confidence interval of the other (better performing) strategies.

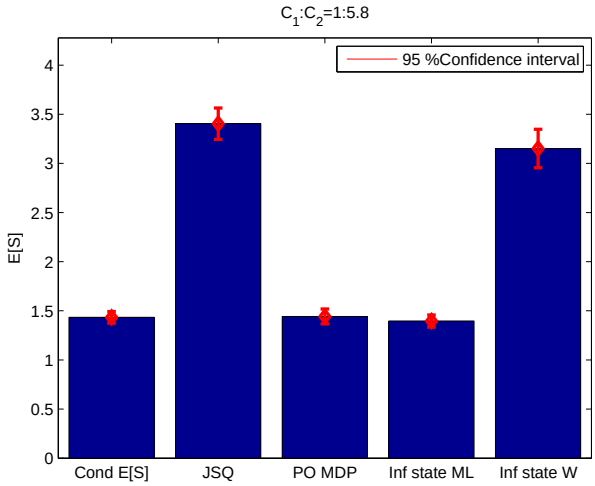


Figure 38: OPNET results for $C_1 : C_2 = 1 : 5.79$.

Table 17: Fraction of foreground files routed to node 1 for $C_1 : C_2 = 1 : 5.79$.

Cond E[S]	JSQ	PO MDP	Inf state ML	Inf state W
0.0942	0.6675	NA	0.138	0.4932

In Figure 38 both **Inf state W** and **JSQ** perform considerably worse. What can be observed in Tables 15, 16, 17 is that for the best performing strategies the fraction of foreground files that is routed to node 1 is very low when the capacity of node 1 is much lower than node 2 while for **JSQ** and **Inf state W** this fracion remains high. In the figures can also be observed that the strategies **Cond E[S]**, **PO MDP** and **Inf state ML** remain in each others 95% confidence interval bounds. For these policies the three simulation scenarios did not result in any significant difference in performance. This is also supported by theoretical difference in performance found in Section 4.5.

6 Conclusion

This master project was about finding optimal dynamical strategies for concurrent access in wireless LANs and to determine the impact of burstiness of the arriving traffic on the optimal strategies. In chapter 2 MDP models have been developed that optimize the optimal sojourn time using the observed number of flows on each wireless node. This involved the application of a partial observation model based on the conditioned time limiting distribution over the system for a given dynamic strategy. The models were extended with MMPP (Markov Modulated Poisson Processes) enabling the research on burstines in the traffic arrival processes. The experiments indicate that:

- When burstiness increases ($D \downarrow$ or $T \uparrow$) the preference for the lowest loaded node increases. This can be observed as the movement of the 'switching curve' to the direction of the lowest loaded node. This movement is bigger when the foreground traffic arrival intensity λ_0 increases.
- When λ_0 decreases the 'switching curve' moves to the lowest loaded server.
- The impact of the filesize distribution is small when the squared coefficient of variation c^2 is equal for both background and foreground files. This should however be examined by applying a larger statespace and assymmetric server capacities.
- When the squared coefficient of variation of foreground traffic c_0^2 grows larger, with fixed background squared coefficient of variation $c_1^2 = c_2^2 < c_0^2$, the switching curve moves to the lowest loaded node.

The partial observation approach in Section 2.3 does not take in account the path or history of observations until a certain number of flows is reached. The history of observations provide extra information about the distribution of flows between foreground traffic and background traffic. Therefore in Chapter 3 Bayesian dynamic programming have been studied. With Bayesian dynamic programming the statespace is extended to a probability distribution over the statespace. The distribution is called an information state and represents the belief on the real system state based on the observation history. Every observation the distribution is updated. On this distribution a MDP can be defined. Solving this MDP is complicated. First discretization has to be applied on the statespace. Even with course discretization, the statespace with Bayesian dynamic programming grows very large, even with course discretization. Therefore a combination of the full observable MDP and the information state distribution is proposed. Two variants have been considered:

- Weighted, with this variant all decisions of the fully observable MDP solution are weighted on the information state. The decision closest to this weighted decision is selected.
- Maximum likelihood, with this variant the most likely state is chosen from the information state distribution. The optimal decision corresponding tho that state is selected.

In the MDPs the cost function represents the expected number of flows. The actual optimization objective is however the expected sojourn time. For this

purpose an algortihm have been developed that can calculate the sojourn time, conditioned on the number of flows, for dynamic strategy on the number of flows. For a simple PS queue the expression for the sojourn time conditioned on the number of flows is known. From this single PS conditional sojourn time expresion a simple decision strategy has been developed: the 'conditional sojourn time strategy'. This strategy has been compared against the fully observable MDP. The expected sojourn time of the 'conditional sojourn time split' has a very small difference to expected sojourn time of the fully observable MDP strategy.
All the proposed strategies in this thesis has been implemented in OPNET, a realistic simulation environment that implements the full range of lower-layer protocol details. For the two combinations of the full observable MDP and the information state, proposed in Chapter 3, the results indicated that the weighted variant performed very poor compared to the maximum likelihood variant. The maximum likelihood variant performed in the range of the partially observable MDPs and the conditional sojourn time strategy. The conditional sojourn time split performance is in the range of the best performing strategies together with the MDP based strategies. A big advantage of the conditional sojourn time strategy is that this strategy can be calculated on line and no algorithm has to be applied off line.

7 Topics for further research

During the development of the different strategies that are described in this thesis new ideas came for extending these strategies. The time for working on this thesis was however limited. These ideas can be a start on continuing research on dynamic server selection strategies for concurrent access. The ideas involve:

- Impact of difference in burstiness on the policies generated by the MDPs in Section 2.6. All flows in the experiments in Section 2.7 have equal burstiness parameters. What if these parameters differ?
- Improve conditional sojourn time heuristic for high foreground load on networks with symmetrical capacity. In the case of high foreground load an symmetrical capacity the performance of the conditional sojourn time heuristic differs from the full observable MDP solution. This is because the heuristic does not use the foreground arrival rate for the optimization.
- Conditional sojourn heuristic for MMPP arrivals and H_2 file size distribution. The conditional sojourn time heuristic is based on the assumption of Poisson arrivals and exponential file sizes. This heuristic can be extended with MMPP arrivals and H_2 file size distribution.
- Parameter learning (Q-learning). In practice there are cases where the actual traffic parameters are not exactly known and have to be measured. Models can be formulated that will (robustly) adapt the necessary parameters automatically.

8 Appendix

8.A Markov Decision Processes

8.A.1 Semi-Markov Decision Processes

A Semi-Markov Decision Process consists of the combination $(S, \mathcal{A}, p, r)$ where

$\mathcal{S}$ is the state space,

$\mathcal{A}$ is the set of decisions that can be made,

$p(s, a, s')$ with $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ is the transition matrix for the embedded Markov Chain of the system, and

$r(s, a)$ is the reward or cost for each state and decision.

Furthermore, for each state and decision, the transition times are modeled as random variables $T(s, a)$ with expectation $\tau(s, a) = \mathbb{E}[T(s, a)]$. Let $R(s)$ be policy, which is a set of decisions $a \in \mathcal{A}$ for each state $s \in \mathcal{S}$. So a policy is a dynamical decision strategy with decisions that are based on the state space $\mathcal{S}$. For a given policy $R(s)$ the embedded Markov chain defined by $p(s, a, s')$ can be solved. Define π_R as the solution for the embedded Markov chain defined by $p(s, R(s), s')$. From the embedded Markov chain the time limiting distribution ν_R can be calculated using the expectations of the transition times $\tau(s, a)$:

$$\nu_R(s) = \frac{\pi_R(s)\,\tau(s, R(s))}{\sum_{s' \in \mathcal{S}} \pi_R(s)\,\tau(s', R(s'))}. \tag{70}$$

Define g as the total long term expected reward. Using the time limiting distribution ν_R, g can be calculated by:

$$g = \sum_{s \in \mathcal{S}} \nu_R(s)\, r(s, R(s)). \tag{71}$$

See [1] for more information about continuous-time semi-Markov decision processes.

8.A.2 Value iteration

A Semi-Markov Decision Process can be optimized by finding an optimal policy R^*, which is the set of optimal decisions when the system is in state s, that minimizes the total long term expected reward g. The optimal policy R^* can be obtained by applying backwards recursion. For a backwards recursion algorithm it is necessary that all the expected transition times $\tau(s, a)$ are equal. Then for each iteration the expected epoch or transition length is equal. This can be done by adding dummy transitions that will not change the state but will equilize the expected transition times for all states. First note that the optimal policy depends only on $T(s, a)$ through its expectation $\tau(s, a)$, independent of the distribution of $T(s, a)$ [1] (see Equation (71)). So $T(s, a)$ can be any distribution with expectation $\tau(s, a)$. Now choose $T(s, a) = \tau G(s, a)$ with $0 < \tau \leq \tau(s, a), \forall s \in \mathcal{S}$ and $G(s, a)$ a geometric distribution with parameter $q(s, a) = \frac{\tau}{\tau(s,a)}$. Because the epochs are lower than or equal to the

expected transition time virtual or dummy transitions can occur when the system state does not change. The number of dummy transitions before the system state changes has a geometric distribution. $G(s,a)$ has an expectation that is equal to $\mathbb{E}[G(s,a)] = \frac{1}{q(s,a)}$ and so $\mathbb{E}[T(s,a)] = \tau\mathbb{E}[G(s,a)] = \tau(s,a)$. With $T(s,a) = \tau G(s,a)$ additional dummy transitions are added with probability $1 - q(s,a)$. For Semi-Markov Decision Processes the backward recursion is defined by [1]:

$$V_{(t+1)\tau}(s) = \max_{a\in\{1,2\}} \left\{ r(s,a)\tau + q(s,a) \sum_{s'\in\mathcal{S}} p(s,a,s') V_{t\tau}(s') + (1 - q(s,a)) V_{t\tau}(s) \right\}. \tag{72}$$

This backwards recursion will further be denoted with the name 'value iteration'.

8.A.3 Uniformization

A special case is when the distribution of transition times $T(s,a)$ is exponential with rates $\lambda(s,a,s'), s,s' \in \mathcal{S}$. Such a process is called a continuous-time Markov Decision Process (MDP). Backward recursion for MDPs can be formulated in a different way. Note that that the minimum of two independent exponentially distributed variables $X \sim exp(\lambda), Y \sim exp(\mu)$ is again exponentially distributed: $\min\{X,Y\} \sim exp(\lambda+\mu)$. Using this property for each state dummy transitions $\lambda(s,a,s) = \gamma - \sum_{s\neq s'} [\lambda(s,a,s')]$ can be added in order let the sum of the rates out of each state equal

$$\gamma \geq \sum_{s'\in\mathcal{S}} \lambda(s,a,s'), \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{73}$$

When in all states the sum of rates out is equal, the expected time until the next transition $\tau(s,a)$ is also equal for all states. The expected time untill the next transition will be $\frac{1}{\gamma}$. The concept of making the rates out of states equal is called uniformization. With the uniformized Markov chain the transition rates can be transformed in transition probabilities $p(x,y) = \frac{\lambda(x,y)}{\gamma}$ that can be used in the backward recursion. Since the rates out are equal, the expected transition times $q(s,a)$ can be chosen $q(s,a) = 1$. [1].

8.B IPP as renewal process

In the MMPP cookbook [11] it is written that an IPP is stochastically equivalent to a Hyperexponential renewal process. The corresponding density function of the interarrival times is

$$\begin{aligned} f_{H_2}(t) &= p\mu_1 e^{-\mu_1 t} + (1-p)\mu_2 e^{-\mu_2 t}, \\ p &= \frac{\lambda - \mu_2}{\mu_1 - \mu_2}. \end{aligned} \tag{74}$$

Before this transformation can be derived some knowledge about the distribution of MMPP is needed. The transition probability matrix for a MMPP

renewal sequence is given in [11]:

$$\begin{aligned} F(t) &= \int_0^t e^{(Q-\Lambda)u} du \Lambda \\ &= \left\{ I - e^{(Q-\Lambda)t} \right\} (\Lambda - Q)^{-1} \Lambda \\ &= \left\{ I - e^{(Q-\Lambda)t} \right\} F(\infty). \end{aligned} \tag{75}$$

From the transition probability matrix the density can be determined by differentiating the transition probability matrix:

$$f(t) = \left\{ -(Q-\Lambda) e^{(Q-\Lambda)t} \right\} F(\infty). \tag{76}$$

This expression contains a matrix exponential which is the Taylor series expansion for the exponential with matrix algebra. The calculation can be simplified by using diagonalization. When a matrix A is diagonalized it is decomposed into PDP^{-1} where D is the diagonal matrix with eigenvalues and P is the matrix with eigenvectors with the same ordering as the corresponding eigenvalues. Thus the decomposition will become $\begin{bmatrix} \psi_1 & \psi_2 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \psi_1 & \psi_2 \end{bmatrix}^{-1}$ with ψ_i the eigenvectors and λ_i the eigenvalues. The nice property of this diagonalization is that the matrix exponential can be simplified:

$$\begin{aligned} A &= PDP^{-1} \\ A^k &= PD^k P^{-1}, \\ e^{At} &= \sum_{k=0}^{\infty} \frac{(At)^k}{k!} = P \sum_{k=0}^{\infty} \frac{(Dt)^k}{k!} P^{-1} = P e^{Dt} P^{-1}. \end{aligned} \tag{77}$$

For $Q - \Lambda$ the diagonalization will become:

$$\begin{aligned} PDP_{-1} &= Q - \Lambda \\ e^{(Q-\Lambda)t} &= P e^{Dt} P^{-1} \\ \det\left[(Q-\Lambda) - \gamma I \right] &= \lambda\omega_2 + \gamma(\omega_2 + \omega_1 + \lambda) + \gamma^2, \\ \mu_1 &= \frac{1}{2}\left(\omega_2 + \omega_1 + \lambda - \sqrt{(\omega_2 + \omega_1 + \lambda) - 4\omega_2\lambda} \right), \\ \mu_2 &= \frac{1}{2}\left(\omega_2 + \omega_1 + \lambda + \sqrt{(\omega_2 + \omega_1 + \lambda) - 4\omega_2\lambda} \right), \\ \gamma &= -\mu_1 \vee -\mu_2, \\ D &= \begin{bmatrix} -\mu_1 & 0 \\ 0 & -\mu_2 \end{bmatrix}, \\ e^{Dt} &= \begin{bmatrix} e^{-\mu_1 t} & 0 \\ 0 & e^{-\mu_2 t} \end{bmatrix}, \\ P &= \begin{bmatrix} 1 & 1 \\ \frac{\mu_2}{\mu_2 - \lambda} & \frac{\mu_1}{\mu_1 - \lambda} \end{bmatrix}. \end{aligned} \tag{78}$$

Now $F(\infty)$ will be calculated:

$$\begin{aligned} F(\infty) &= (\Lambda - Q)^{-1}\Lambda \\ &= \frac{1}{\lambda}\begin{bmatrix}\omega_2 & \omega_1 \\ \omega_2 & \lambda+\omega_1\end{bmatrix}\cdot\begin{bmatrix}\lambda & 0 \\ 0 & 0\end{bmatrix} \\ &= \begin{bmatrix}1 & 0 \\ 1 & 0\end{bmatrix}. \end{aligned} \tag{79}$$

Using the diagonalization $f(t)$ can be expressed in terms of diagonal exponentials:

$$\begin{aligned} f(t) &= \left\{-(Q-\Lambda)e^{(Q-\Lambda)t}\right\}F(\infty) \\ &= \left\{-PDP^{-1}Pe^{Dt}P^{-1}\right\}\begin{bmatrix}1 & 0 \\ 1 & 0\end{bmatrix} \\ &= \left\{-PDe^{Dt}P^{-1}\right\}\begin{bmatrix}1 & 0 \\ 1 & 0\end{bmatrix}. \end{aligned} \tag{80}$$

Using basic linear algebra operations the transition density matrix can be obtained:

$$\begin{aligned} f(t) &= \left\{-PDe^{Dt}P^{-1}\right\}\begin{bmatrix}1 & 0 \\ 1 & 0\end{bmatrix} \\ &= \begin{bmatrix}p_{1,1} & p_{1,2} \\ p_{2,1} & p_{2,2}\end{bmatrix}\begin{bmatrix}\mu_1^{-\mu_1 t} & 0 \\ 0 & \mu_2 e^{-\mu_2 t}\end{bmatrix}\frac{1}{|P|}\begin{bmatrix}p_{2,2} & -p_{1,2} \\ -p2,1 & p_{1,1}\end{bmatrix}\begin{bmatrix}1 & 0 \\ 1 & 0\end{bmatrix} \\ &= \begin{bmatrix}p_{1,1} & p_{1,2} \\ p_{2,1} & p_{2,2}\end{bmatrix}\frac{1}{|P|}\begin{bmatrix}(p_{2,2}-p_{1,2})\mu_1^{-\mu_1 t} & 0 \\ (p_{1,1}-p_{2,1})\mu_2 e^{-\mu_2 t} & 0\end{bmatrix} \\ &= \frac{1}{|P|}\begin{bmatrix}p_{1,1}(p_{2,2}-p_{1,2})\mu_1 e^{-\mu_1 t}+p_{1,2}(p_{1,1}-p_{2,1})\mu_2^{-\mu_2 t} & 0 \\ p_{2,1}(p_{2,2}-p_{1,2})\mu_1 e^{-\mu_1 t}+p_{2,2}(p_{1,1}-p_{2,1})\mu_2^{-\mu_2 t} & 0\end{bmatrix} \end{aligned} \tag{81}$$

$$\begin{aligned} \frac{1}{|P|} &= \frac{1}{\frac{\mu_1(\mu_2-\lambda)-\mu_2(\mu_1-\lambda)}{(\mu_1-\lambda)(\mu_2-\lambda)}} \\ &= \frac{(\mu_1-\lambda)(\mu_2-\lambda)}{\lambda(\mu_1-\mu_2)}, \end{aligned} \tag{82}$$

$$\begin{aligned} p_{1,1}(p_{2,2}-p_{1,2}) &= \frac{\lambda}{\mu_1-\lambda}, \\ p_{1,2}(p_{1,1}-p_{2,1}) &= -\frac{\lambda}{\mu_2-\lambda}, \\ f_{1,1}(t) &= \frac{(\mu_1-\lambda)(\mu_2-\lambda)}{\lambda(\mu_1-\mu_2)}\left(\frac{\lambda}{\mu_1-\lambda}\mu_1 e^{-\mu_1 t}-\frac{\lambda}{\mu_2-\lambda}\mu_2^{-\mu_2 t}\right) \\ &= \left(\frac{\mu_1-\lambda}{\mu_1-\mu_2}\mu_1 e^{-\mu_1 t}+\frac{\lambda-\mu_2}{\mu_1-\mu_2}\mu_2^{-\mu_2 t}\right). \end{aligned} \tag{83}$$

$f_{1,1}(t)$ corresponds to the interarrival time because, due to the nature of IPP only arrivals can occur when the arrival rate is 'on'.

8.C Partial observation simulation program

```
lambda0 = .5;
lambda1 = .3;
lambda2 = .3;
mu1 = 1;
mu2 = 1;
gamma = lambda0 + lambda1 + lambda2 + mu1 + mu2;
time = 0;
alphaMat; %Poisson arrivals exponential file sizes
%full observable MDP solution for given parameters.

x1=0;
z1=0;
z1old=0;
z2old=0;
x2=0;
z2=0;
v1=zeros(1,n+1);%next information state
u1=zeros(1,n+1);%current information state
v1(1) = 1;
u1(1) = 1;
v2=zeros(1,n+1);%next information state
u2=zeros(1,n+1);%current information state
v2(1) = 1;
u2(1) = 1;

tmax = 500000;%#iterations

x=zeros(tmax,2);%sequence of real # foreground flows
z=zeros(tmax,2);%observed # of flows
%sequence of information states on node 1 and 2:
vseq1=zeros(tmax,n+1);
vseq2=zeros(tmax,n+1);

while time < tmax
    %only update if observed state changes
    if(z1old≠z1 || z2old≠z2)
        x(time + 1,:) = [x1 x2];
        z(time + 1,:) = [z1 z2];
        z1old = z1;
        z2old = z2;
        vseq1(time+1,:)=v1;
        vseq2(time+1,:)=v2;
        time = time + 1;
        u1=zeros(1,n+1);
        u2=zeros(1,n+1);
        for i=0:z1;
            u1(i+1) = v1(i+1);
        end;
        for i=0:z2;
            u2(i+1) = v2(i+1);
        end;
    end

    v1=zeros(1,n+1);
    v2=zeros(1,n+1);
    rnd = rand;
    if(rnd < (lambda0 / gamma));
        rnd = rand();
        if(alphaMat(z1−x1+1,x1+1,z2−x2+1,x2+1) == 0 || ...
```

```
            alphaMat(z1-x1+1,x1+1,z2-x2+1,x2+1) == 1 && rnd < .5);
        for i=0:z1;
            v1(i+2)=u1(i+1);
        end;
        for i=0:z2;
            v2(i+1)=u2(i+1);
        end;
        x1 = x1 + 1;
        z1 = z1 + 1;
    elseif(alphaMat(z1-x1+1,x1+1,z2-x2+1,x2+1) == 2 || ...
            alphaMat(z1-x1+1,x1+1,z2-x2+1,x2+1) == 1 && rnd > .5)
        for i=0:z1;
            v1(i+1)=u1(i+1);
        end;
        for i=0:z2;
            v2(i+2)=u2(i+1);
        end;
        x2 = x2 + 1;
        z2 = z2 + 1;
    end;
elseif(rnd < ((lambda0 + lambda1) / gamma));
    for i=0:z1;
        v1(i+1)=u1(i+1);
    end;
    for i=0:z2;
        v2(i+1)=u2(i+1);
    end;
    if(z1 < n);
        z1 = z1 + 1;
    end;
elseif(rnd < ((lambda0 + lambda1 + mu1) / gamma));
    if(z1 > 0);
        for i=0:z1-1;
            v1(i+1)=u1(i+2)*(i+1)/z1+u1(i+1)*(z1-i)/z1;
        end;
        z1 = z1 - 1;
        rnd = rand();
        if(rnd < x1 / (z1 + 1E-12) && x1 > 0);
            x1 = x1 - 1;
        end;
    else
        for i=0:z1;
            v1(i+1) = u1(i+1);
        end;
    end;
    for i=0:z2;
        v2(i+1)=u2(i+1);
    end;
elseif(rnd < ((lambda0 + lambda1 + mu1 + lambda2) / gamma));
    for i=0:z1;
        v1(i+1)=u1(i+1);
    end;
    for i=0:z2;
        v2(i+1)=u2(i+1);
    end;
    if(z2 < n);
        z2 = z2 + 1;
    end;
elseif(rnd < ((lambda0 + lambda1 + mu1 + lambda2 + mu2) / gamma));
    for i=0:z1;
        v1(i+1)=u1(i+1);
    end;
```

```
        if(z2 > 0);
            for i=0:z2-1;
                v2(i+1)=u2(i+2)*(i+1)/z2+u2(i+1)*(z2-i)/z2;
            end;
            z2 = z2 - 1;
            rnd = rand();
            if(rnd < x2 / (z2 + 1E-12) && x2 > 0);
                x2 = x2 - 1;
            end;
        else
            for i=0:z2;
                v2(i+1) = u2(i+1);
            end;
        end;
    end;
end;
```

References

[1] S. Bhulai and G. Koole. Lecture notes stochastic optimization. Lecture Notes, February 2008.

[2] S.C. Borst, O.J. Boxma, and N. Hegde. Sojourn times in finite-capacity processor-sharing queues. In *Proceedings NGI 2005 Conference*, 2005.

[3] O. J. Boxma. Stochastic performance modelling, February 2002. 11.

[4] M. H. DeGroot. *Optimal Statistial Decisions*. McGraw-Hill, 1975.

[5] G.J.Hoekstra and R.D. van der Mei. On the processor sharing of file transfers in wireless lans. In *Proceedings of the 69th IEEE Vehicular Technology Conference, VTC Spring 2009, 26-29 April 2008, Barcelona, Spain*. IEEE, 2009.

[6] OPNET Technologies Inc. Opnet modeler, May 2009. http://www.opnet.com/solutions/network_rd/modeler.html.

[7] R. Litjens, F. Roijers, J.L. Van den Berg, R.J. Boucherie, and M.J. Fleuren. Performance analysis of wireless LANs: An integrated packet/flow level approach. In *Proceedings of the 18th International Teletraffic Congress - ITC18*, pages 931–940, Berlin, Germany, 2003.

[8] U. Rieder. Bayesian dynamic programming. *Advances in Applied Probability*, 7:330–348, June 1975.

[9] B. Sengupta and D.L. Jagerman. A conditional response time of the m/m/1 processor sharing queue. *AT&T Technical Journal*, 64(2):409–421, February 1985.

[10] H. C. Tijms. *A first course in stochastic models*. John Wiley & Sons, Ltd., 2003.

[11] K. Meier-Hellstern W. Fisher. The markov-modulated poisson process (mmpp) cookbook. *Performance Evaluation*, 18:149–171, 1992.

[12] Y. Wu, C. Williamson, and J. Luo. On processor sharing and its applications to cellular data network provisioning. *Performance Evaluation*, 64(9-12):892–908, 2007.