VU University Amsterdam
Faculty of Sciences
Business Mathematics and Informatics

Research Paper

Safar C. Niamat

Parallel Processor Sharing Systems: Concurrent Access in Wireless Networks

Supervisor: Dr. Sandjai Bhulai

Amsterdam
January 2011

Abstract

The number of active mobile phone users has increased tremendously in the past years. Research shows that data services will grow more and more and that voice services will lose their weight in the overall user bill. With limited bandwidth and the growing need for data services wireless network providers are left with a dilemma how to keep offering their users an acceptable Quality of Service with a low-bit cost. This problem becomes an even bigger issue in densely populated areas in which there are many more active users per network.

Concurrent Access (CA) may offer a solution for this problem. Given the fact that any given area is covered by multiple mobile networks and the upcoming use of dual-sim dual-antenna mobile phones create the possibility for CA. These types of phones make it possible to use two different networks simultaneously. Different CA models are described in this paper. The server selection models give arriving foreground jobs access to just one of the two networks. The static server selection models use a pre-defined probability to decide to which network an arriving foreground job should be sent, while the dynamic server selection models use information available of both networks to decide where an arriving foreground job should be sent. The job split models split an arriving foreground job in two parts and sends each part to a network. The static job split models use a pre-defined split factor, while the dynamic job split models use available information from both networks to decide the split factor. Only the server selection models and the static job split models will be analyzed in this paper.

Contents

1. Introduction 3
1.1. Recent developments in wireless telecommunication 3
1.2. Purpose of the thesis 4
1.3. Structure of the thesis 5
2. Concurrent access models 6
2.1. Static server selection model 6
2.2. Dynamic server selection model 7
2.3. Static job split model 8
2.4. Dynamic job split model 9
3. Numerical analysis of the CA models 11
3.1. Static server selection model 11
3.2. Dynamic server selection model 13
3.3. Static job split model 14
4. Comparison of the CA models 17
4.1. Static server selection model versus dynamic server selection model 17
4.2. Static server selection model versus static job split model 18
4.3. Dynamic server selection model versus static job split model 19
5. Conclusion 22
Bibliography 23

1. Introduction

This chapter gives an overview of the development of wireless telecommunication technologies from the very first beginning till this moment. The purpose and structure of this thesis will also be mentioned later on.

1.1. Recent developments in wireless telecommunication

On 10 March 1876 Alexander Graham Bell made the first telephone call in the history. Since then many developments have taken place in this area. The first-generation (1G) wireless telephone technology was deployed in the late 1970s and could only transmit analogue voice information. The 2G wireless phones used Global System for Mobile Communications (GSM) and were first used in the early 1990s in Europe. This system provided limited data services (such as fax and SMS) and improved the quality of the audio by using digital modulation schemes. The 2.5G wireless phones used General Packet Radio Service (GPRS), which enhanced the data capacity of GSM and mitigated some of its limitations. This enabled fast transmissions of text and graphics-rich data as packets. The Wireless Application Protocol (WAP) technology allowed these phones to access Web pages. Enhanced Data Rates for GSM Evolution (EDGE) is an expansion of GPRS technology, which uses existing GSM-networks, and increases the amount of data that can be transmitted per unit of time. The 3G wireless phones are known as Universal Mobile Telecommunications Systems (UMTS or IMT2000) and sustain higher data rates and open the door to many Internet applications. The technology adds multimedia facilities to 2G phones by allowing video, audio and graphics applications. High-Speed Downlink Packet Access (HSDPA) can be seen as the technology for 3.5G wireless mobile phones, which can have a transmission speed that is 10 times the speed of UMTS and allows broadband wireless connections (ADACHI (2001), ASHIHO (2003), WIKIPEDIA).

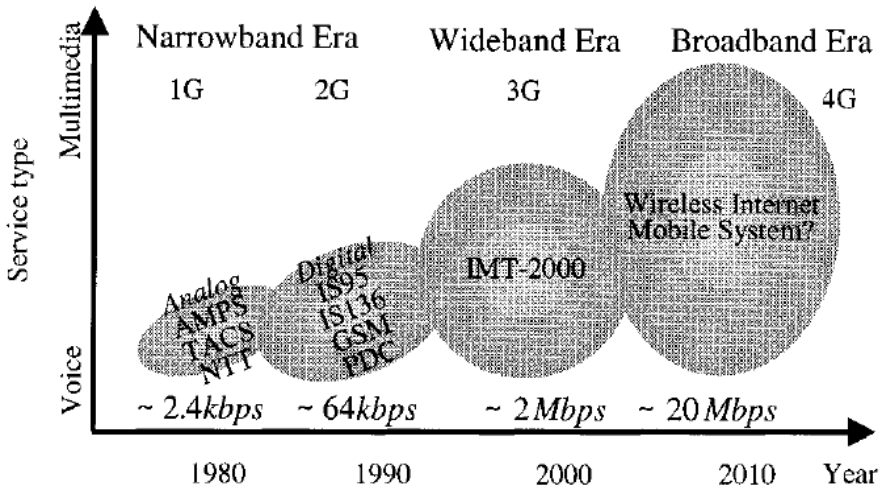

Figure 1.1: Development of wireless networks (ADACHI (2001)).

HSDPA can be seen as a step towards 4G wireless networks. Figure 1.1 shows the development of wireless networks in time and the growing development of multimedia services. Much research is done on what the main objectives of 4G wireless networks should

be. Research shows that there is consensus that 4G networks should have a low-bit cost which is essential because it is expected that the data services will grow more and more and that voice services will lose their weight in the overall user bill. The new technology should also be available to any user, anywhere, anytime and it should have a multi-service platform, because that is the main reason for user transactions and will give telecommunication operators access to new levels of traffic (PEREIRA AND SOUSA). Table 1.1 shows the different major services mobile network providers offer their users as a result of the growing technology and the ever changing needs of users from one generation to another.

Generation	1G	2G	3G	4G
Wireless Access	Analog	*Digital*	*Digital*	*Digital*
	FDMA	TDMA, *CDMA*	*CDMA*	*?*
Major Services	Voice	Voice	Voice	*Voice over IP*
		Internet (text only)	*Internet (text, images)*	*Rich Internet*
Core-network	Circuit-based	Circuit-based	Circuit- and *Packet-based*	*Fully IP-based*

Table 1.1: Differentiation of 1G to 4G systems (ADACHI (2001)).

1.2. Purpose of the thesis

The number of mobile networks users is growing rapidly, which leads to more usage of the networks. The reason for this is the decreasing purchasing costs for mobile phones, which makes mobile phones accessible for more people. The technological development brings many possibilities with it and leads to more intensive usage and larger download requests. These developments lead to a higher load on the existing networks, which form a threat for services that make intensive use of the network (e.g., mobile television, video conference, online gaming, etc.). As a result of this, the performance perceived by the users will be less than accepted in most of the cases, because it will take a lot of time for these services to be transmitted. This becomes a major problem in areas that have many users of specific wireless networks, since there are many more requests per time unit in these areas, a network could get overloaded.

Wireless networks often overlap each other in terms of coverage, especially in areas that are densely populated. Dual-sim dual-antenna phones are able to use multiple wireless networks simultaneously (see Figure 1.2), which creates the possibility to get better performances in these areas. Concurrent Access (CA) is a new promising technique which provides solutions to wireless network providers to increase the performance perceived by their users in these densely populated areas by increasing the available amount of bandwidth. CA is a much cheaper solution to improve the performance of wireless networks than purchasing more frequencies and thus a very attractive solution for wireless network providers. **Although this is a very active research area, little applicable results have been published so far!** Initial results were obtained by GUNAWAN (2008), however, the results are not mature enough to be of practical use. This paper will extend their results and attempt to surpass their achievements.

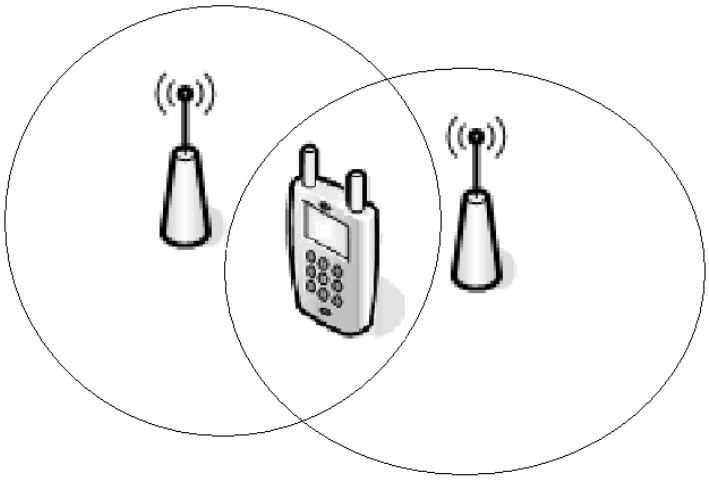

Figure 1.2: Mobile phone that can use multiple networks simultaneously in an overlapping coverage area between different wireless network providers.

The main objective of this paper is to study what the benefits are of CA in wireless networks by using CA models and evaluating them. This will be done by using mathematical models that represent these networks and by analyzing these models. The download requests will be modeled as foreground jobs, the requested download time as the required service time and the actual download time as the sojourn time. All other types of requests will be modeled as background jobs; this simulates the behavior of the other users which use the networks (users that only use one network).

1.3. Structure of the thesis

The different CA-models will be described first, looking at their characteristics, strengths and weaknesses. The differences between the models will be analyzed and the impact that these differences can have in finding an optimal minimum sojourn time for foreground jobs. The goal of all the models is to improve the QoS of the foreground jobs. Past papers on Concurrent Access will be studied and their results and findings will be mentioned. The results of the different models will be given and analyzed. These results will be compared with each other and the impact of the different models on the minimum sojourn times of foreground jobs will be pointed out. The paper will end with a conclusion and recommendations for future research in this area.

2. Concurrent access models

There are different models that can be used for CA: Static Server Selection, Dynamic Server Selection, Static Job Split and Dynamic Job Split. The first two models select the server to which a complete job should go, while the last two models split a job in two parts and send each part to one of the servers. The objectives are to find the optimal server selection probability in the server selection models and an optimal split factor in the job split models. The goal of these models is to minimize the mean sojourn time of the foreground jobs (type 0 jobs). By decreasing the mean sojourn time the networks can guarantee better performances to their users.

CA models are queuing models with two processor sharing servers and three types of jobs (0, 1, and 2). Jobs of type 0 (foreground jobs) can access both servers, while the other jobs j (background jobs) can only access server j (j = 1, 2).

The notations used for the models are:
- i: the type of job {0, 1, 2};
- j: the number of the server {1, 2};
- λ_i: arrival rate of type i jobs;
- β_i: service time of jobs of type i jobs;
- $\rho_i = \beta_i * \lambda_i$: occupation rate of type i jobs;
- ρS_j = the occupation rate of server j.

The assumptions of the models are:
- The arrival of the different types of jobs is a Poisson process;
- The service times are exponentially distributed;
- The capacity of the servers are equal (C_1=C_2).

The following condition is necessary to keep the system stable:

$$\rho S_j < 1, \qquad \text{for } j = 1, 2. \tag{2.1}$$

2.1. Static server selection model

Current mobile phones can only use one specific wireless network (see Figure 2.1). In processor sharing systems jobs share the capacity of a server. Thus a 1 minute job which uses the full capacity of the server will be processed in 1 minute, but when there are two 1 minute jobs arriving at the same time at a server, both jobs will take 2 minutes to be processed because they have to share the server's capacity (each job uses 50% of the server's capacity).

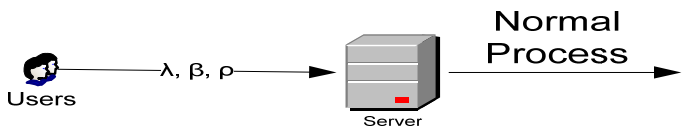

Figure 2.1: A normal processor sharing system.

The performance perceived by mobile phone users is acceptable when the network has a low occupancy rate (ρ), which means that there is enough capacity available for new download requests. This is the case in areas with a small number of active users of a specific network. In densely populated areas there are many more active users per network, which lead to poor performances for the users of these networks.

In densely populated areas there are also many overlapping wireless networks. These overlapping areas give the opportunity to switch to other networks or to use two different networks at the same time for dual-sim dual-antenna phones. The Static Server Selection model can offer a solution for these areas (see Figure 2.2). Every time a foreground job is transmitted, it is sent to server 1 with a probability q and to server 2 with a probability (1-q). This model is beneficial when there are networks with high occupancy rates and low occupancy rates overlapping each other. The model basically creates a joint capacity: the capacity which is not used in both servers $C_{joint} = (C_1 - \rho_1) + (C_2 - \rho_2)$ will be used as one capacity to process all foreground jobs. The main goal of the model is to minimize the mean sojourn time of the foreground jobs.

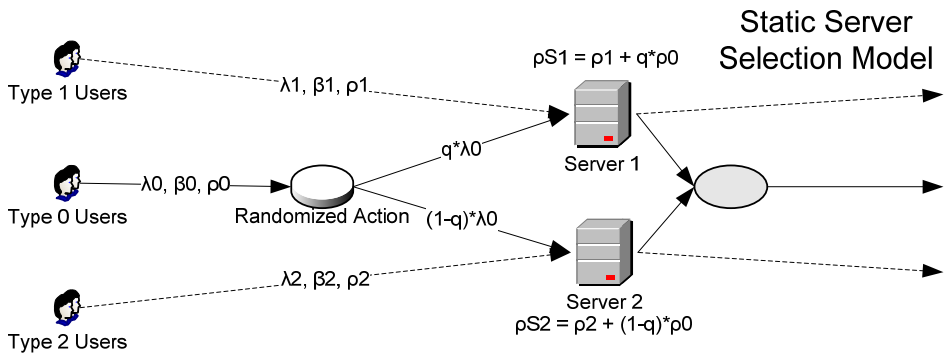

Figure 2.2: In the Static Server Selection Model the background jobs i can only use server i, while the foreground jobs are sent to one of the 2 servers depending on the probability q (Randomized Action).

To satisfy condition **(2.1)** for a stable system the following conditions should hold:

$$\rho S_1 = \rho_1 + q\rho_0 < 1 \text{ and } \rho S_2 = \rho_2 + (1-q)\rho_0 < 1. \qquad \textbf{(2.2)}$$

2.2. Dynamic server selection model

In the Static Server Selection model the probability q is pre-defined. This means that every foreground job goes to server 1 with a pre-defined probability q. Let us assume that q > (1-q), thus more foreground jobs are sent to server 1 than to server 2. It could happen that at certain moments server 1 is occupied more than server 2. It would be more beneficial to send more jobs to server 2 at these moments, but since the q is pre-defined the model will keep sending most of the foreground jobs to server 1. This is not optimal at all! The Dynamic Server Selection model offers a solution to this problem (see Figure 2.3). The model uses the information available from both servers at the moment a foreground job is transmitted and then it decides to which server the job should be sent. The model will make better decisions at the Decision Rule when it has more information available. BHULAI et al. (2010) show that

the dynamic model with a Bayesian decision methodology gives mean sojourn times which are not significantly worse than a model with all information. The Bayesian methodology is also better applicable to a multi-network environment, since it cannot be assumed that a system can observe all information.

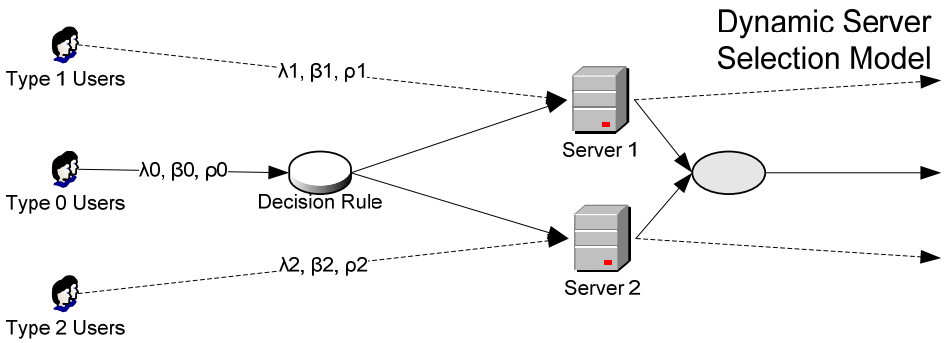

Figure 2.3: In the Dynamic Server Selection Model the background jobs i can only use server i, while the foreground jobs are sent to one of the 2 servers depending on the information of both servers (Decision Rule).

Condition **(2.1)** is also necessary in this model to keep the system stable. To satisfy condition **(2.1)** the following condition should hold in each server at any time:

$$\frac{1}{2}\sum_{1}^{2}\beta_i\lambda_i < 1. \qquad \textbf{(2.3)}$$

2.3. Static job split model

The Static Job Split model has a different approach in reducing the mean sojourn times of foreground jobs than the previous models. The model splits each foreground job in two parts and sends each part to a server (see Figure 2.4). The previous models split the arrival stream of foreground jobs (λ_0) in two parts, while this model splits the service time (β_0) in two parts. Both models attempt to reduce the load of each server, but differ in the way they attempt to reduce it. Let us assume that we have the Static Server Selection model with $q = 0.5$, then $\rho S_1 = \beta_1 \lambda_1 + \beta_0 (q\lambda_0) = \rho_1 + 0.5\rho_0$. And let us take a Static Job Split model and use a split factor $\alpha = 0.5$, now we get for $\rho S_1 = \beta_1\lambda_1 + (\alpha\beta_0)\lambda_0 = \rho_1 + 0.5\rho_0$. The Static Jobs Split model is more complicated than the Server Selection models, because each part of a foreground job is dependent on its counterpart. This means that each processed part has to wait for their counterpart to be able to finish completely (this happens at the Multiplexer) and that makes the split model hard to analyze mathematically. Simulation offers a good solution for this problem, but simulations only give robust results when the number of simulated events is high. A multiplexer is a device that combines several input information signals to one output signal (WIKIPEDIA), thus in this case a multiplexer combines 2 parts of data into one data part.

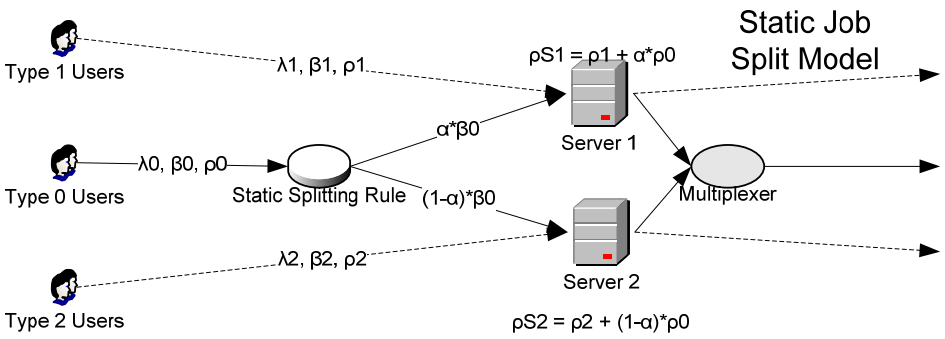

Figure 2.4: In the Static Job Split Model the background jobs i can only use server i, while the foreground jobs are split in two parts α and (1-α) which are sent to server 1 and 2 (Static Splitting Rule), respectively. Finished parts wait for their counterparts and are merged as a whole job (Multiplexer).

Condition **(2.1)** is also necessary in this model to keep the system stable. To satisfy condition **(2.1)** the following condition should hold in each server at any time:

$$\rho S_1 = \rho_1 + \alpha\rho_0 < 1 \text{ and } \rho S_2 = \rho_2 + (1- \alpha)\rho_0 < 1. \quad \textbf{(2.4)}$$

2.4. Dynamic job split model

In the Static Job Split model the split factor α is also pre-defined, meaning that every arriving foreground job is split the same way and each part is sent to the server it was pre-defined to be sent to. Let us now assume that α > (1- α), thus a larger part of foreground jobs are sent to server 1 than to server 2. It could happen that at certain moments server 1 is occupied more than server 2. It would be more beneficial in these cases to sent a larger part of the foreground jobs to server 2, but since α is pre-defined the model will keep sending larger parts of foreground jobs to server 1. This is also not optimal! The Dynamic Job Split model offers a solution to this problem (see Figure 2.5). The model will use information available from both servers at the moment a foreground job arrives and it will decide what size of the foreground job to sent to server 1 and 2 (Dynamic Splitting Rule). The Dynamic Server Selection model can be seen as a simple version of the Dynamic Job Split model, because the Dynamic Server Selection model sends only complete jobs to a server every time a foreground job arrives, thus α is either 1 or 0. There are no studies or results of the Dynamic Job Split model in this area so far. The model is very complicated to implement and there is no such criteria for the Dynamic Splitting Rule. A successful Dynamic Splitting Rule must reduce the time a specific part of a foreground job has to wait for its counter part at the multiplexer, because the only way to minimize the mean sojourn time of foreground jobs will be by minimizing the time parts of foreground jobs have to wait at the multiplexer. This Dynamic Splitting Rule should take into consideration the information available of both servers at the moment a foreground job arrives (e.g., number of jobs, processing times of the jobs in each server) and the possible changes that could take place in the near future (the load of the background jobs). For example: if a foreground job arrives, it could be split in two parts depending on just the information available in both servers at that moment, but it could be that a less occupied server 1 at that moment could become more occupied just after those foreground job parts have been sent to the servers, because the less occupied server 1 has a higher background

traffic. This will result in a higher waiting time for the part in server 2 for its counterpart in server 1, which was expected to be processed faster.

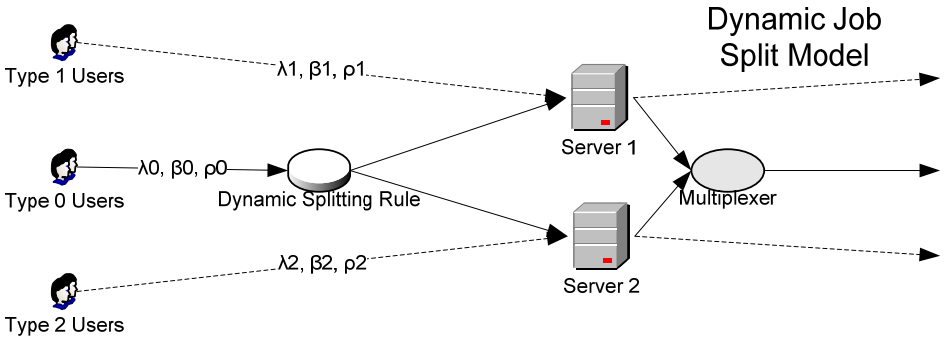

Figure 2.5: In the Dynamic Job Split Model the background jobs i can only use server i, while the foreground jobs are split in two parts for each server. In the Dynamic model the split factor depends on the current status of both servers (Dynamic Splitting Rule). Earlier finished parts have to wait for their counterparts (Multiplexer).

Condition **(2.1)** is also necessary in this model to keep the system stable. To satisfy condition **(2.1)**, condition **(2.3)** should hold in each server at any time.

All the different CA models have been described in this chapter. It is clear now how the process of a system containing one of these models work. In the next chapter the Server Selection models and the Static Job Split model will be analyzed. This will be done by analyzing the results that these models give and comparing these results with each other in Chapter 4.

3. Numerical analysis of the CA models

In this chapter all the results of the Concurrent Access models will be analyzed for each model. Only three intensities of foreground traffic have been analyzed: low, medium and high traffic, which contain loads of respectively 0.1, 0.5 and 0.9.

3.1. Static server selection model

The Static Server Selection model is a very simple model, which can be solved mathematically. GUNAWAN (2008) shows how the optimal q (q*) can be calculated mathematically. The results of q* for $\rho_0 = 0.1$, $\rho_0 = 0.5$ and $\rho_0 = 0.9$ are given in Tables 3.1. Only the values above the main diagonal are calculated, because the values under the main diagonal can easily be calculated using the following formula:

$$q(\rho_1,\rho_2)^* = 1 - q(\rho_2,\rho_1). \quad \textbf{(3.1)}$$

All the values on the main diagonal of the tables are 0.5, because both servers have equal occupancy rates and that is why the jobs are sent to each server with equal probability. The system is not stable for all values of ρ_1 and ρ_2, these are the colored areas in the tables. Note: $\rho_0 + \rho_1 + \rho_2 \geq 2$ in all the colored areas. It clearly shows that the selection probability decreases when ρ_1 increases for any value of ρ_2, thus more jobs are sent to server 2 when the occupation rate of server 1 increases.

0.9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.172	0.500
0.8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.202	0.500	
0.7	0.000	0.000	0.000	0.000	0.000	0.000	0.215	0.500		
0.6	0.000	0.000	0.000	0.000	0.000	0.223	0.500			
0.5	0.000	0.000	0.000	0.000	0.228	0.500				
0.4	0.000	0.000	0.000	0.231	0.500					
0.3	0.000	0.000	0.234	0.500						
0.2	0.000	0.236	0.500							
0.1	0.237	0.500								
0	0.500									
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.1 (a): Values of q*. $\rho_0 = 0.1$.

0.9	0.000	0.000	0.000	0.035	0.084	0.138				
0.8	0.000	0.016	0.067	0.121	0.180	0.245	0.317			
0.7	0.034	0.088	0.144	0.204	0.269	0.338	0.414	0.500		
0.6	0.103	0.160	0.220	0.283	0.351	0.422	0.500			
0.5	0.172	0.231	0.294	0.359	0.427	0.500				
0.4	0.240	0.301	0.365	0.431	0.500					
0.3	0.307	0.369	0.433	0.500						
0.2	0.372	0.435	0.500							
0.1	0.437	0.500								
0	0.500									
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.1 (b): Values of q*. $\rho_0 = 0.5$.

0.9	0.0577	0.0833								
0.8	0.1192	0.1510	0.1852							
0.7	0.1760	0.2113	0.2489	0.2894						
0.6	0.2292	0.2667	0.3064	0.3488	0.3945					
0.5	0.2794	0.3183	0.3593	0.4029	0.4495	0.5000				
0.4	0.3272	0.3670	0.4088	0.4530	0.5000					
0.3	0.3729	0.4133	0.4556	0.5000						
0.2	0.4168	0.4575	0.5000							
0.1	0.4591	0.5000								
0	0.5000									
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.1 (c): Values of q*. $\rho_0 = 0.9$.

After deriving the optimal selection probability (q*) the expected sojourn times can be calculated by using the following formula from GUNAWAN (2008):

$$E[S_0] = \sum_{i=1}^{2} q_i \frac{\beta_0}{1 - q_i \rho_0 - \rho_i}. \qquad \textbf{(3.2)}$$

Tables 3.2 show the results of the expected minimum sojourn times of the model given the optimal selection probability q* and $\beta_0 = 1$. There are no mean sojourn times for those areas where the system is unstable. Only the values of the mean sojourn times above the main diagonal of the tables are calculated, because you can easily derive the values of the bottom part by mirroring the table:

$$E[S](\rho_1,\rho_2) = E[S](\rho_2,\rho_1). \qquad \textbf{(3.3)}$$

The minimum mean sojourn time is 1.053 when both servers only serve foreground jobs, thus even if there is no background traffic the mean sojourn time of the foreground jobs will be more than β_0. The maximum mean sojourn time is equal to 20 and this is the case when:

$$\rho_1 + \rho_2 = 1.9 - \rho_0 \quad \text{given that } \rho_1 = \rho_2. \qquad \textbf{(3.4)}$$

Thus when both servers have equal loads and the used capacity is equal to 1.9. This is the maximum capacity that can be used to keep the system stable when using 1 decimal digits for all ρ. These systems are processor sharing systems, which mean that all jobs in a server have to share the capacity of the server. Thus the mean sojourn time increases when the traffic of any job type increases. In Table 3.2 (a) we see that the mean sojourn time does not change in certain cases, when the traffic in server 1 increases (see columns of the table). The reason is that the jobs are sent to server 2 to which the traffic is constant.

0.9	1.111	1.250	1.429	1.667	2.000	2.500	3.333	5.000	9.142	20.000
0.8	1.111	1.250	1.429	1.667	2.000	2.500	3.333	4.747	6.667	
0.7	1.111	1.250	1.429	1.667	2.000	2.500	3.214	4.000		
0.6	1.111	1.250	1.429	1.667	2.000	2.430	2.857			
0.5	1.111	1.250	1.429	1.667	1.954	2.222				
0.4	1.111	1.250	1.429	1.635	1.818					
0.3	1.111	1.250	1.405	1.538						
0.2	1.111	1.232	1.333							
0.1	1.096	1.176								
0	1.053									
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.2 (a): Values of the minimum E[S]. $\beta_0 = 1$, $\rho_0 = 0.1$, Min E[S] = 1.05, Max E[S] = 20.

0.9	2.000	2.500	3.333	4.861	7.899	16.944				
0.8	2.000	2.495	3.200	4.242	5.952	9.325	19.314			
0.7	1.989	2.398	2.933	3.666	4.743	6.497	9.928	20.000		
0.6	1.922	2.250	2.661	3.194	3.919	4.972	6.667			
0.5	1.828	2.093	2.412	2.809	3.318	4.000				
0.4	1.726	1.939	2.190	2.490	2.857					
0.3	1.622	1.795	1.993	2.222						
0.2	1.521	1.662	1.818							
0.1	1.425	1.538								
0	1.333									
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.2 (b): Values of the minimum E[S]. $\beta_0 = 1$, $\rho_0 = 0.5$, Min E[S] = 1.33, Max E[S] = 20.

0.9	7.403	15.556								
0.8	5.535	8.603	17.778							
0.7	4.432	6.071	9.332	19.072						
0.6	3.700	4.722	6.412	9.768	19.776					
0.5	3.174	3.870	4.903	6.605	9.975	20.000				
0.4	2.776	3.277	3.968	4.989	6.667					
0.3	2.463	2.837	3.327	4.000						
0.2	2.208	2.496	2.857							
0.1	1.997	2.222								
0	1.818									
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.2 (c): Values of the minimum E[S]. $\beta_0 = 1$, $\rho_0 = 0.9$, Min E[S] = 1.82, Max E[S] = 20.

3.2. Dynamic server selection model

For the Dynamic Server Selection model it is not needed to calculate an optimal selection probability, because the foreground jobs are sent to each server depending on information available about each server. BHULAI et al. (2010) use two different types of Decision Rules: one is based on a fully observed system and the other is based on a Bayesian model. The paper shows that the Bayesian model gives robust results which are not significantly worse than the fully observed system. The Bayesian model is also better applicable to a multi-network environment as it cannot be assumed that a system is fully observed.

Tables 3.3 give the results of the fully observed system, which is just a little better than the system based on a Bayesian model. The minimum mean sojourn time is 1.08 when both of the background traffic have load of 0.1. The maximum mean sojourn time is around 11 and this is also the case when **(3.4)** holds.

0.9		1.241	1.409	1.625	1.911	2.308	2.897	3.858	5.705	11.020
0.8		1.228	1.383	1.574	1.813	2.125	2.542	3.138	4.056	
0.7		1.210	1.350	1.514	1.711	1.952	2.255	2.641		
0.6		1.189	1.311	1.452	1.613	1.799	2.020			
0.5		1.164	1.271	1.390	1.519	1.665				
0.4		1.139	1.232	1.331	1.437					
0.3		1.115	1.195	1.277						
0.2		1.092	1.160							
0.1		1.072								
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.3 (a): Values of the minimum E[S]. $\beta_0 = 1$, $\rho_0 = 0.1$, Min E[S] = 1.08, Max E[S] = 11.02.

0.9		2.212	2.750	3.615	5.280	9.721				
0.8		1.991	2.371	2.924	3.817	5.552	10.279			
0.7		1.814	2.096	2.485	3.047	3.954	5.699	10.854		
0.6		1.666	1.886	2.171	2.552	3.114	4.013			
0.5		1.542	1.716	1.931	2.209	2.584				
0.4		1.437	1.579	1.749	1.960					
0.3		1.353	1.468	1.603						
0.2		1.278	1.374							
0.1		1.216								
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.3 (b): Values of the minimum E[S]. $\beta_0 = 1$, $\rho_0 = 0.5$, Min E[S] = 1.22, Max E[S] = 10.86.

0.9		8.907								
0.8		5.064	9.104							
0.7		3.696	5.228	9.633						
0.6		2.938	3.796	5.447	10.163					
0.5		2.458	2.993	3.864	5.599	10.615				
0.4		2.126	2.495	3.032	3.923					
0.3		1.883	2.152	2.513						
0.2		1.697	1.900							
0.1		1.553								
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.3 (c): Values of the minimum E[S]. $\beta_0 = 1$, $\rho_0 = 0.9$, Min E[S] = 1.55, Max E[S] = 10.62.

3.3. Static job split model

A simulation was used for the Static Job Split model, because the dependencies of the parts of the foreground jobs make it very difficult to solve the problem mathematically. To get robust results many events should be simulated. 100 000 job events of type 0 have been simulated for the results obtained in the tables below (Tables 3.4 and 3.5). The best method to determine α^* would be to smoothen the graph with all the split factors and determine the minimum mathematically. This method was not very applicable because the graphs had many outliers which influenced the graph a lot and gave values for α^* that were not the optimal split factor. The only way to solve this problem would be to simulate more job events, but this would take

days to simulate and the improvement would not be significant. To avoid this problem it was decided to use three iterations in getting an optimal split factor with 3 decimal digits. The first iteration would give the first decimal value, which would be used to get the second decimal value in the next iteration by zooming into the area around the value found in the previous iteration and so on. The value of β_0 was set to 60 to get the following results for α^* (Tables 3.4). For a fixed value of ρ_2 it shows that α^* decreases when the value of ρ_1 is increased, thus the part of the foreground jobs that is sent to server 1 becomes smaller when server 1 has a higher occupancy rate. The optimal split factor for $\rho_1 = \rho_2$, has not been calculated, because this is always equal to 0.5. To following formula can be used to determine the values of α^* below the main diagonal:

$$\alpha\,(\rho_1,\rho_2)^* = 1 - \alpha\,(\rho_2,\rho_1). \qquad \textbf{(3.5)}$$

0.9	0.056	0.045	0.047	0.046	0.061	0.072	0.089	0.144	0.289	
0.8	0.047	0.054	0.054	0.067	0.075	0.151	0.260	0.341		
0.7	0.098	0.098	0.106	0.137	0.122	0.237	0.346			
0.6	0.133	0.156	0.195	0.185	0.244	0.294				
0.5	0.195	0.242	0.245	0.307	0.444					
0.4	0.306	0.326	0.354	0.418						
0.3	0.345	0.435	0.482							
0.2	0.496	0.497								
0.1	0.497									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.4 (a): Values of α^*. $\rho_0 = 0.1$.

0.9	0.071	0.062	0.084	0.130	0.133	0.128				
0.8	0.075	0.112	0.132	0.197	0.254	0.305	0.352			
0.7	0.150	0.167	0.225	0.275	0.265	0.362	0.445			
0.6	0.232	0.255	0.244	0.326	0.342	0.442				
0.5	0.300	0.331	0.344	0.399	0.445					
0.4	0.329	0.385	0.377	0.405						
0.3	0.405	0.433	0.449							
0.2	0.445	0.485								
0.1	0.497									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.4 (b): Values of α^*. $\rho_0 = 0.5$.

0.9	0.104	0.095								
0.8	0.155	0.218	0.212							
0.7	0.251	0.238	0.260	0.328						
0.6	0.250	0.300	0.300	0.372	0.390					
0.5	0.295	0.336	0.378	0.400	0.461					
0.4	0.378	0.399	0.409	0.453						
0.3	0.430	0.438	0.482							
0.2	0.431	0.465								
0.1	0.495									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.4 (c): Values of α^*. $\rho_0 = 0.9$.

Tables 3.5 give the values that were found as the optimal mean sojourn times given the values of α^* from the tables above. The minimum mean sojourn time is 35.34, which is almost 60% of β_0. It is very remarkable that we can achieve mean sojourn times lower than β_0 with the Job Split model, but this is only the case when we have plenty processing capacity available. The mean sojourn times increase a lot when there is more traffic. The maximum mean sojourn time is almost 22 times bigger than β_0.

0.9	74.69	77.98	89.17	109.08	137.55	187.51	277.21	419.99	663.34	
0.8	63.58	72.11	83.11	100.02	125.81	172.99	241.46	340.99		
0.7	62.07	68.89	79.37	94.55	117.95	152.36	192.91			
0.6	58.43	65.38	75.07	87.13	106.76	136.85				
0.5	55.19	60.49	67.91	77.05	91.21					
0.4	50.34	54.77	60.58	69.06						
0.3	45.90	50.18	53.92							
0.2	40.42	43.37								
0.1	35.34									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.5 (a): Values of the minimum E[S]. $\beta_0 = 60$, $\rho_0 = 0.1$, Min E[S] = 35.34, Max E[S] = 663.34.

0.9	138.35	177.32	249.05	344.57	562.60	1031.67				
0.8	132.70	162.97	216.23	262.29	398.82	615.65	945.33			
0.7	114.50	143.11	175.59	221.44	329.35	418.04	615.69			
0.6	101.83	124.18	150.91	189.62	256.39	331.42				
0.5	90.05	106.20	124.51	155.34	190.15					
0.4	79.06	92.56	105.84	130.47						
0.3	68.70	77.16	88.64							
0.2	58.85	65.43								
0.1	49.32									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.5 (b): Values of the minimum E[S]. $\beta_0 = 60$, $\rho_0 = 0.5$, Min E[S] = 49.32, Max E[S] = 1031.67.

0.9	548.11	1072.17								
0.8	433.27	815.96	867.22							
0.7	323.98	391.80	632.29	1301.90						
0.6	249.90	297.57	430.60	627.19	1059.59					
0.5	210.16	252.66	324.10	452.42	601.50					
0.4	173.54	194.35	244.23	336.07						
0.3	138.13	155.12	195.61							
0.2	109.58	126.02								
0.1	87.16									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 3.5 (c): Values of the minimum E[S]. $\beta_0 = 60$, $\rho_0 = 0.9$, Min E[S] = 87.16, Max E[S] = 1301.9.

Results of all the models were shown in this chapter for low, medium and high traffic. In the next chapter the results will be compared to decide which model is the best.

4. Comparison of the CA models

In this chapter all the results from the previous chapter will be compared by looking at the relative differences.

4.1. Static server selection model versus dynamic server selection model

To compare the different server selection models the difference between the optimal sojourn times of both models are compared. The relative difference is shown in Tables 4.1. The following formula was used to calculate the relative differences:

$$\Delta = \frac{E[S_{StaticServerSelection}] - E[S_{DynamicServerSelection}]}{E[S_{StaticServerSelection}]} \times 100\,. \qquad \textbf{(4.1)}$$

There exists no such scenario where the Static Server Selection model outperforms the Dynamic Server Selection model. For any fixed value of ρ_1 the tables show that the relative difference increases a lot for higher values of ρ_2. The Dynamic Server Selection model performs much better than the Static Server Selection model when both servers have high occupancy rates. The Dynamic Server Selection model also performs much better when the difference between ρ_1 and ρ_2 becomes smaller. In these cases it is more crucial to decide to which server a foreground job should be sent to. In the cases where the difference between the two background traffic streams is higher the positive impact of the Decision Rule on the minimum mean sojourn times becomes smaller.

0.9		0.7%	1.4%	2.5%	4.4%	7.7%	13.1%	22.8%	37.6%	44.9%
0.8		1.8%	3.2%	5.5%	9.4%	15.0%	23.7%	33.9%	39.2%	
0.7		3.2%	5.5%	9.2%	14.5%	21.9%	29.8%	34.0%		
0.6		4.9%	8.2%	12.9%	19.4%	26.0%	29.3%			
0.5		6.9%	11.0%	16.6%	22.3%	25.1%				
0.4		8.9%	13.7%	18.6%	21.0%					
0.3		10.8%	14.9%	17.0%						
0.2		11.3%	13.0%							
0.1		8.9%								
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 4.1 (a): Relative difference between the Static Server Selection model and the Dynamic Server Selection model. $\beta_0 = 1$, $\rho_0 = 0.1$.

0.9		11.5%	17.5%	25.6%	33.2%	42.6%				
0.8		20.2%	25.9%	31.1%	35.9%	40.5%	46.8%			
0.7		24.3%	28.5%	32.2%	35.7%	39.1%	42.6%	45.7%		
0.6		25.9%	29.1%	32.1%	34.9%	37.4%	39.8%			
0.5		26.3%	28.8%	31.2%	33.4%	35.4%				
0.4		25.9%	27.9%	29.8%	31.4%					
0.3		24.6%	26.4%	27.8%						
0.2		23.1%	24.4%							
0.1		21.0%								
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 4.1 (b): Relative difference between the Static Server Selection model and the Dynamic Server Selection model. $\beta_0 = 1$, $\rho_0 = 0.5$.

0.9		42.7%								
0.8		41.1%	48.8%							
0.7		39.1%	44.0%	49.5%						
0.6		37.8%	40.8%	44.2%	48.6%					
0.5		36.5%	38.9%	41.5%	43.9%	46.9%				
0.4		35.1%	37.1%	39.2%	41.2%					
0.3		33.6%	35.3%	37.2%						
0.2		32.0%	33.5%							
0.1		30.1%								
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 4.1 (c): Relative difference between the Static Server Selection model and the Dynamic Server Selection model. $\beta_0 = 1$, $\rho_0 = 0.9$.

4.2. Static server selection model versus static job split model

The same comparison method was used for comparing the Static Server Selection model with the Static Job Split model. The minimum sojourn times of the foreground jobs of the Static Server Selection was re-calculated using formula **(3.2)** with $\beta_0 = 60$. These new values for the foreground jobs of the Static Server Selection model were compared with those of the Static Job Split model. Tables 4.2 show the relative difference between the models using the following formula:

$$\Delta = \frac{E[S_{StaticServerSelection}] - E[S_{StaticJobSplit}]}{E[S_{StaticServerSelection}]} \times 100 . \qquad \textbf{(4.2)}$$

All the negative values are bold colored in the tables. These negative values mean that the Static Server Selection model outperforms the Static Job Split model. The tables clearly show that the Static Job Split model performs better when the background traffic is low, but when the background traffic increases the model is outperformed by the Static Server Selection model. Still there are some cases where the Static Job Split model does better than the Static Server Selection model when both servers have high background traffic. The Static Server Selection model is a simple version of the Static Job Split model, thus the Static Job Split model should be able to perform better in all cases or at least just as good. This is not always

the case, because of the negative effect the dependency of the job parts have on the sojourn times in the Static Job Split model. Better results could be gained by simulating the Static Job Split model for a longer period.

0.9	**-12.0%**	**-4.0%**	**-4.0%**	**-9.1%**	**-14.6%**	**-25.0%**	**-38.6%**	**-40.0%**	**-20.9%**	
0.8	4.6%	3.9%	3.0%	**0.0%**	**-4.8%**	**-15.3%**	**-20.7%**	**-19.7%**		
0.7	6.9%	8.1%	7.4%	5.4%	1.7%	**-1.6%**	**0.0%**			
0.6	12.4%	12.8%	12.4%	12.9%	11.0%	6.1%				
0.5	17.2%	19.3%	20.8%	23.0%	22.2%					
0.4	24.5%	27.0%	29.3%	29.6%						
0.3	31.2%	33.1%	36.0%							
0.2	39.4%	41.3%								
0.1	46.3%									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 4.2 (a): Relative difference between the Static Server Selection model and the Static Job Split model. $\beta_0 = 60$, $\rho_0 = 0.1$.

0.9	**-15.3%**	**-18.2%**	**-24.5%**	**-18.1%**	**-18.7%**	**-1.5%**				
0.8	**-10.6%**	**-8.9%**	**-12.6%**	**-3.1%**	**-11.7%**	**-10.0%**	18.4%			
0.7	4.0%	0.5%	0.2%	**-0.7%**	**-15.7%**	**-7.2%**	**-3.4%**			
0.6	11.7%	8.0%	5.5%	1.1%	**-9.0%**	**-11.1%**				
0.5	17.9%	15.4%	14.0%	7.8%	4.5%					
0.4	23.7%	20.5%	19.5%	12.7%						
0.3	29.4%	28.4%	25.9%							
0.2	35.5%	34.4%								
0.1	42.3%									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 4.2 (b): Relative difference between the Static Server Selection model and the Static Job Split model. $\beta_0 = 60$, $\rho_0 = 0.5$.

0.9	**-23.4%**	**-14.9%**								
0.8	**-30.5%**	**-58.1%**	18.7%							
0.7	**-21.8%**	**-7.6%**	**-12.9%**	**-13.8%**						
0.6	**-12.6%**	**-5.0%**	**-11.9%**	**-7.0%**	10.7%					
0.5	**-10.3%**	**-8.8%**	**-10.2%**	**-14.2%**	**-0.5%**					
0.4	**-4.2%**	1.2%	**-2.6%**	**-12.3%**						
0.3	6.5%	8.9%	2.0%							
0.2	17.3%	15.8%								
0.1	27.3%									
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 4.2 (c): Relative difference between the Static Server Selection model and the Static Job Split model. $\beta_0 = 60$, $\rho_0 = 0.9$.

4.3. Dynamic server selection model versus static job split model

The results of the Dynamic Server Selection model were obtained with $\beta_0 = 1$ and the results of the Static Server Selection were simulated using $\beta_0 = 60$. The mean sojourn times of both

models are divided by the β_0 that was used to gain their results. These values give the growth of the sojourn times compared to the original service time (β_0). This relationship will be compared for the Dynamic Server Selection model and the Static Job Split model using the following formula:

$$\Delta = \frac{E[S_{DynamicServerSelection}] \times \mu_{DynamicServerSelection} - E[S_{StatcJobSplit}] \times \mu_{StatcJobSplit}}{E[S_{DynamicServerSelection}] \times \mu_{DynamicServerSelection}} \times 100 . \qquad \textbf{(4.3)}$$

Note that $1/\beta$ is equal to μ. The bold colored values in Tables 4.3 give all the cases where the Dynamic Server Selection model outperforms the Static Job Split model. It shows again that the Job Split model performs better when there is less traffic. It is very remarkable that the Dynamic Server Selection model performs better in most cases when $\rho S_1 + \rho S_2 \geq 1$. The dependency of the parts of the Static Job Split model has a bigger negative impact on the minimum sojourn times when the servers have higher occupancy rates. The parts will take longer to process and parts that have been finished will have to wait longer for their counterparts.

0.9		**-4.7%**	**-5.5%**	**-11.9%**	**-19.9%**	**-35.4%**	**-59.5%**	**-81.4%**	**-93.8%**	
0.8		2.1%	**-0.2%**	**-5.9%**	**-15.7%**	**-35.7%**	**-58.3%**	**-81.1%**		
0.7		5.1%	2.0%	**-4.1%**	**-14.9%**	**-30.1%**	**-42.6%**			
0.6		8.3%	4.5%	**0.0%**	**-10.3%**	**-26.8%**				
0.5		13.4%	10.9%	7.6%	**-0.1%**					
0.4		19.9%	18.1%	13.5%						
0.3		25.0%	24.8%							
0.2		33.8%								
0.1										
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 4.3 (a): Relative difference between the relationship of the Dynamic Server Selection model and the Static Job Split model with their β_0. $\rho_0 = 0.1$.

0.9		**-33.6%**	**-50.9%**	**-58.8%**	**-77.6%**	**-76.9%**				
0.8		**-36.5%**	**-52.0%**	**-49.5%**	**-74.1%**	**-84.8%**	**-53.3%**			
0.7		**-31.5%**	**-39.6%**	**-48.5%**	**-80.1%**	**-76.2%**	**-80.0%**			
0.6		**-24.2%**	**-33.3%**	**-45.6%**	**-67.5%**	**-77.4%**				
0.5		**-14.8%**	**-20.9%**	**-34.0%**	**-43.5%**					
0.4		**-7.3%**	**-11.7%**	**-24.3%**						
0.3		5.0%	**-0.6%**							
0.2		14.6%								
0.1										
0										
ρ1/ρ2	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**

Table 4.3 (b): Relative difference between the relationship of the Dynamic Server Selection model and the Static Job Split model with their β_0. $\rho_0 = 0.5$.

0.9		-100.6%								
0.8		-168.5%	-58.8%							
0.7		-76.7%	-101.6%	-125.3%						
0.6		-68.8%	-89.1%	-91.9%	-73.8%					
0.5		-71.3%	-80.5%	-95.1%	-79.0%					
0.4		-52.4%	-63.2%	-84.7%						
0.3		-37.3%	-51.5%							
0.2		-23.8%								
0.1										
0										
ρ1/ρ2	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9

Table 4.3 (c): Relative difference between the relationship of the Dynamic Server Selection model and the Static Job Split model with their β_0. $\rho_0 = 0.9$.

All the models have been compared in this chapter. The Static Job Split model clearly outperforms both models when the total traffic in both servers is smaller than 1. But the Static Job Split model performs worse than the Static Server Selection model when the total traffic in both server increases. The Dynamic Server Selection model outperforms the Static Server Selection model and gives better results than the Static Job Split model when the traffic in both servers is equal to or more than 1. The next chapter gives a final conclusion regarding the whole paper.

5. Conclusion

The goal of this paper was to explore the possibilities of Concurrent Access in existing wireless networks. Concurrent access is the answer to increase bandwidth in current wireless networks by combining multiple existing networks to create a joint capacity, which will increase the Quality of Service for download requests of foreground jobs. This is a low cost improvement with a high gain in the existing wireless networks.

The Dynamic Server Selection model had by far the best performance, especially when there is high traffic in both servers. The Static Job Split model performs best when there is low traffic in both servers. The dependency of the parts of the foreground jobs has a negative impact when minimizing the sojourn times of the foreground jobs; this impact becomes bigger when the traffic in the servers increases. The best solution for now could be a hybrid method, which uses the Static Job Split model in cases where the total traffic is less than 1 and uses the Dynamic Server Selection model in cases where the total traffic is equal to or more than 1.

The Dynamic Job Split model has the potential to outperform the Static Job Split model and the Dynamic Server Selection model, because it has more possibilities: it can send complete jobs or parts of it to servers and the model uses information available about both servers when making splitting decisions. The goal of this model should be to find an algorithm that will minimize the negative effect the dependency of the parts has on the sojourn time of foreground jobs. This should be done by taking into consideration the information available in both servers and information about the background traffic of each server.

This paper leaves many interesting topics for future research:

- Implementation of the Dynamic Job Split model;
- Models with minimum QoS restrictions for the background jobs;
- Models with servers that have unequal capacity;
- Models which can use more than 2 servers simultaneously;
- Models with arrivals that are not according a Poisson process;
- Models with service times that are not exponentially distributed;
- How to implement the models of this paper in a real environment.

Bibliography

1. ADACHI, F. (2001). *Wireless Past and Future - Evolving Mobile Communications Systems, IEICE Trans. Fundamentals (Vol. E84-A, NO. 1)*, January 2001. Found at: http://www.mobile.ecei.tohoku.ac.jp/intro/pdf/12_IEICE_2001_adachi.pdf
2. ASHIHO, L.S. (2003). *Mobile Technology: Evolution from 1G to 4G*, June 2003. Found at: http://www.electronicsforu.com/EFYLinux/efyhome/cover/jun2003/Mobile-tech.pdf
3. PEREIRA, V. AND SOUSA. *T. Evolution of Mobile Communications: from 1G to 4G,* Department of Informatics Engineering of the University of Coimbra.
4. GUNAWAN, R. (2008). *Concurrent Access to Mobile Networks*, VU University Amsterdam, May 2008.
5. BHULAI, S., HOEKSTRA, G.J., BOSMAN, J.W., VAN DER MEI, R.D. (2010). *Dynamic Traffic Splitting to Parallel Wireless Networks with Partial Information: A Bayesian Approach*, VU University Amsterdam, CWI and Thales, September 2010.
6. WIKIPEDIA

http://www.telecomwereld.nl/mobgesch.htm (19-1-2011)
http://www.phonehistory.co.uk/mobile-phones-timeline.html (19-1-2011)
http://nl.wikipedia.org/wiki/Universal_Mobile_Telecommunications_System (17-4-2010)
http://searchmobilecomputing.techtarget.com/generic/0,295582,sid40_gci1078079,00.html (Unknown)