Practical 1

1) Investigating an e-commerce site

a.) I use the Utilisation Law to calculate the utilisation and the total service demand that is needed for the Bottleneck analysis. Here is the analysis of the whole system and the formulas I used:

$$D_i = \frac{U_i}{X}$$
$$U_i = X_i \cdot S_i = \lambda_i \cdot S_i$$
$$X = \lambda$$

Var	WS	AS	A	B	C	$CPU1$	$CPU2$
λ_i	λ	λ	6λ	4.5λ	3λ	5λ	5λ
S_i(in s)	0.110	0.230	0.060	0.056	0.068	0.020	0.020
U_i	0.11λ	0.23λ	0.36λ	0.252λ	0.204λ	0.1λ	0.1λ
D_i	0.11	0.23	<u>0.36</u>	0.252	0.204	0.1	0.1

. Even trough I should analysis the database server only, I can obtain the bottleneck of the database server also form this data. Consequently disk A is the bottleneck of the database server.

b.) The minimum average residence time is defined as the sum of the single minimum residence times, that are simple given by: $n + i * S_i$ because we assume no waiting time. (General Residence Time Law)

$$\begin{aligned}\overline{W}_{min} &= \sum_i W_{i_{min}} \\ &= W_A + W_B + W_C + W_{P_1} + W_{P_2} \\ &= 4 \cdot 60 + 3 \cdot 56 + 2 \cdot 68 + 5 \cdot 20 + 5 \cdot 20 = 744ms\end{aligned}$$

c.) The maximum utilisation of disk C can occur if the database server if full utilised, that is limited by the bottleneck device disc A. We have to consider the case that the incoming rate is high enough to reach a utilisation of 100% disk on disk A. I use the Forced Flow Law to come from the single component to the compound and vice versa.

$$\begin{aligned}\lambda_{A_{max}} &= \frac{1}{S_A} = 16.\overline{6} \\ \lambda_{DBS_{max}} &= \frac{\lambda_{A_{max}}}{4} = 4.1\overline{6} \\ \lambda_{C_{max}} &= \lambda_{DBS_{max}} \cdot 2 = 8.\overline{3} \\ U_{C_{max}} &= \lambda_{C_{max}} \cdot S_C = 0.56\overline{6}\end{aligned}$$

The maximum utilisation is about 57%.

d.) The throughput at disk C implies a throughput and an incoming rate of transactions of the database. 0.002 accesses per millisecond are 2 accesses per second, from this it follows a minimum $\lambda_{DBS} = 1$ (Forced Flow Law). That means at least $\frac{2}{3}$ business functions per second have to pass the application server. I assume that a user starts to think if he get the result of the previous business function. Therefore the average time between two business functions from one user is $10 + 20 = 30$ second. We get an incoming rate of $\frac{1}{30s}$ business functions per user. The number of users is now:

$$n \geq \frac{\frac{2}{3}}{\frac{1}{30}} = 20$$

(simple ratio equation)) There would be at least 20 users in the system.

e.) Without considering the rest of the system we can easy calculate the throughput of the web server.

$$X_{WS} = \frac{U_{WS}}{S_{WS}} = \frac{0.11}{0.110} = 1$$

If we assume that the responds time per business function is 20 seconds like in part d), then is the number of users :

$$n \geq \frac{1}{\frac{1}{30}} = 30$$

Each user triggers 6 b.functions, therefore each user stays $6 \cdot 30s = 180s$ in the system. One hour has 3600 seconds, that gives us $\frac{3600}{180} = 20$ user slots, in conclusion we have $20 \cdot 30 = \underline{600}$ users per hour.

f.) I use the same assumption as in part d). The incoming rate is now: $\lambda = \lambda_1 \cdot n = \frac{1}{15+10} \cdot 100 = 4$ business functions per second.

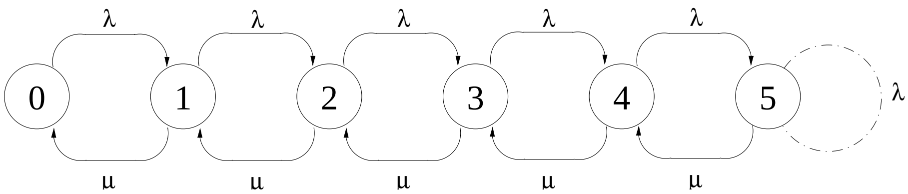
We can use formulas and numbers mentioned in part a). Disk A has a utilisation of $0.36\lambda = 1.44$. I recommend an utilisation of 72%, that implies a double speedup. The service time have to be less than 30 milliseconds. Lets have a look at the other components:

Var	WS	AS	A'	B	C	$CPU1$	$CPU2$
S_i(in s)	0.110	0.230	$\underline{0.030}$	0.056	0.068	0.020	0.020
U_i	0.44	**0.92**	0.72	**1.08**	**0.96**	0.4	0.4

It is strongly recommended to speed up Disk B. Disk C and the application server are high utilised as well.

2) Loss probabilities within a communication network

a.) The transition diagram for the Markov process:



b.) The matrix is:

$$Q = \begin{pmatrix} -\lambda & \lambda & 0 & 0 & 0 & 0 \\ \mu & -(\mu+\lambda) & \lambda & 0 & 0 & 0 \\ 0 & \mu & -(\mu+\lambda) & \lambda & 0 & 0 \\ 0 & 0 & \mu & -(\mu+\lambda) & \lambda & 0 \\ 0 & 0 & 0 & \mu & -(\mu+\lambda) & \lambda \\ 0 & 0 & 0 & 0 & \mu & -\mu \end{pmatrix}$$
$$= \begin{pmatrix} -4 & 4 & 0 & 0 & 0 & 0 \\ 4.8 & -8.8 & 4 & 0 & 0 & 0 \\ 0 & 4.8 & -8.8 & 4 & 0 & 0 \\ 0 & 0 & 4.8 & -8.8 & 4 & 0 \\ 0 & 0 & 0 & 4.8 & -8.8 & 4 \\ 0 & 0 & 0 & 0 & 4.8 & -4.8 \end{pmatrix}$$

c.) I prefer to using `mupad` to solve the linear equation system. The solver expect the input as follows: $\mathcal{M} \cdot x = 0$

```
l:=4:
m:=4.8:
linsolve( { l*p0 - (m+l)*p1 + m*p2 = 0,
  l*p1 - (m+l)*p2 + m*p3 = 0,
  l*p2 - (m+l)*p3 + m*p4 = 0,
  l*p3 - (m+l)*p4 + m*p5 = 0,
  l*p4 - m*p5 = 0,
  p0+p1+p2+p3+p4+p5 = 1
 },
{ p0,p1,p2,p3,p4,p5 }
);

{ p0 = 0.2505881215, p1 = 0.2088234346, p2 = 0.1740195288,
  p3 = 0.145016274,p4 = 0.120846895, p5 = 0.1007057458 }
```

The steady state distribution is:

state	π_0	π_1	π_2	π_3	π_4	π_5
prob	0.251	0.209	0.174	0.145	0.121	0.100

d.) The lost rate can be calculated by multiplying the steady state probability for the state 5 with the incoming rate.

$$\begin{aligned} l &= \lambda \cdot \pi_5 \\ &= 0.4 \end{aligned}$$

There will be lost $0.4\frac{1}{s}$ packets per second, that means every 2.5 seconds one packet will be discarded.

e.)

```
l:=4:
m:=4.8:
P:= linsolve( { l*p0 - (m+l)*p1 + m*p2 = 0,
  l*p1 - (m+l)*p2 + m*p3 = 0,
  l*p2 - (m+l)*p3 + m*p4 = 0,
  l*p3 - (m+l)*p4 + m*p5 = 0,
  l*p4 - (m+l)*p5 + m*p6 = 0,
  l*p5 - m*p6 = 0,
  p0+p1+p2+p3+p4+p5+p6 = 1
 },
{ p0,p1,p2,p3,p4,p5,p6 }
);
 {p0 = 0.2311866052, p1 = 0.1926555044, p2 = 0.1605462536,
p3 = 0.1337885447, p4 = 0.1114904539, p5 = 0.09290871161, p6 = 0.07742392634 }
```

New steady state:

state	π_0	π_1	π_2	π_3	π_4	π_5	π_6
prob	0.231	0.193	0.161	0.134	0.112	0.092	0.077

The lost rate is now: $l \approx 0.31\frac{1}{s}$.

We obtain the average residence in the buffer by multiplying the number of packets before the actual packet with the reciprocal of the release rate. Of course is the state probability to be consider.

$$W_n = \sum_{i=0}^{n} i \cdot \frac{1}{\mu} \cdot \pi_i$$
$$W_5 = 0.412$$
$$W_6 = 0.476$$

The average residence time increase to 1.15% , but the lost rate was decreased to to 80.6%.

```
l:=4:
m:=5.5:
linsolve( ..
```

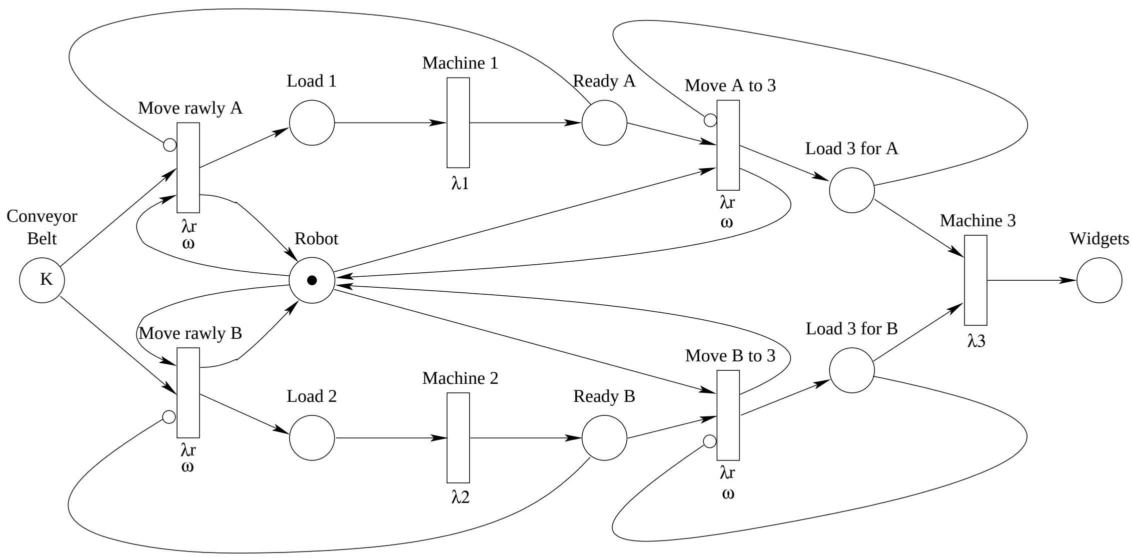
state	π_0	π_1	π_2	π_3	π_4	π_5
prob	0.320	0.233	0.169	0.123	0.090	0.065

The lost rate is now: $l \approx 0.26\frac{1}{s}$ and the residence time is 0.338s
In comparison to the other modifications is the increasing of the bandwidth the most powerful one.

3) Analysing an automated manufacturing system

For each following sub task I used the C-programs that were given and modified them as appropriate.

a.)



variable	λr	$\lambda 1$	$lambda1$	$lambda3$	ω
value	0.01	0.008	0.005	0.01	0.5

The machines are not separately modelled as a place with a token. This model is very easy and specialised.

b.) After editing the c-file to calculate the Utilisation of the machines we achieve the following values.

```
EXPECTED: Utilisation 1: = 0.427696213013
EXPECTED: Utilisation 2: = 0.650098242648
EXPECTED: Utilisation 3: = 0.325049121596
```

The Machine 1/2 is only utilised if transition "t13"/"t23" is enabled, Machine 3 if "t31" is enabled. Of course are this values also available in the standard output file under "Transition - enabled".

c.) The throughput can be read from the "Transition - Average throughput" rows belongs to the machines. We are interesting in machine 3 (t31): $3.250491215960 \cdot 10^{-03} = 0.00325$ widgets per second that is equal to a process time of about $308s$ per widget.

d.) Throughputs:

Buffer:	0	1	2	3	4	5	10
Widgets/s	0.00325	0.00344	0.00355	<u>0.00362</u>	0.00368	0.00372	0.00386

The throughput is dependent from the buffer-size, but the influence of each further buffer cell is decreasing.

e.) There are several possibilities to make the process faster. I used the system with the buffer for further considerations. Here are different suggestions with the resulting values:

Method	Throughput
Machine 2 takes only $150s$	0.00439
Two machines 2 (with $200s$)	0.00412
Robot takes $2s$ for transport	0.00382
New machine 1 (without repairmen)	0.00412
Two old machines 1	0.00415
New machine 1 and $2(150s)$	0.00504

The machine 2 is also a bottleneck. If the machine 1 is still unchanged and the machine 2 can be speeded up or doubled the system will be as fast as a new machine 1. Because the repairmen is also expensive the new machine 1 is a solid and good solution. In my opinion the best solution is to buy a new machine 1 and a new machine 2 (or speed it up). The product rate can be increased to 150%.

f.) The throughput of the system would be 0.00345 widgets per second. The justification is given by the numbers calculated by the SPNP program (simple disable breakdown and setup double service time).