Lab IV: Transportation service dependability

CandID: 10031
Worked with: 10033

Task IV.A - Bus breakdowns

Each bus has a lot of mechanical and electromechanical parts, and firmware and software that need to be working so that the bus can drive. In this task, you are going to study the effect of mechanical failures (hardware) and software failures. All failure and repair times are negative exponentially distributed;
- the time between hardware failures, $T_h \sim n.e.d(\lambda_h)$,
- the time to hardware repair, $T_{rh} \sim n.e.d(\mu_h)$,
- the time between software failures, $T_s \sim n.e.d(\lambda_s)$, and
- the time to software repair, $T_{rs} \sim n.e.d(\mu_s)$.

Task IV.A.1

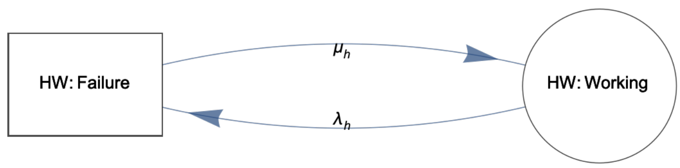
First, consider only hardware failures on a single bus. The objective is to obtain the steady state probabilities that a bus is not working (the unavailability).

a) Define the state variable and events, and draw the Markov model, $\mathcal{M}_h$.
b) What is the unavailability of the bus? Assume $\lambda_h = 1/14$ [once every second week], and $\mu_h = 1$ [repair per day], $\mathcal{P}_h = \{\lambda_h \to 1/14, \mu_h \to 1\}$.
Then, assume that you have $n = 8$ such busses, and with *s=1,2,or 3* repairmen. At least 6 of the 8 busses need to be working in order to serve the passenger demand without too much delays.
c) Define the state variable and events, and draw the Markov model, $\mathcal{M}_c$. What are working states in your model?
d) Use Mathematica to obtain the *steady state probabilities*. Use the parameter set $\mathcal{P}_h$ from above, and change the number of repairmen, *s=1,2,3*.
e) What does *mean time to first failure* (MTFF) mean? What is the MTFF of the bus (in model, $\mathcal{M}_c$)? (for *s=1,2,3*).
f) What does *mean time to failure* (MTTF) mean? What is the MTTF of the bus (in model, $\mathcal{M}_c$)? (for *s=1,2,3*).
g) What does *mean down time* (MDT) mean? What is the MDT of the bus (in model, $\mathcal{M}_c$)? (for *s=1,2,3*).
h) What is *unavailability* and the *expected hours of downtime* in a year the bus system (in model, $\mathcal{M}_c$) is unavailable? (for *s=1,2,3*).
i*) Discuss the different metrics from c) to g) with respect to how useful they are to express the importance of the number of repairmen.

a) In this simple scenario, we can define the state variables 0 and 1, where 0 represents an unavailable bus, and 1 represents an available bus. The events are defined by λ_h and μ_h, where λ_h is the rate of hardware failure, and μ_h is the repair rate. Markov model is illustrated below.

In[ ]:=
```
ClearAll[Q];
Q = Table[",", {2}, {2}];
(Q⟦1, 2⟧ = λh;
  Q⟦2, 1⟧ = μh; ℳh = SetDiagonal[Transpose[Q]]);
(𝒲h = {True, False});
(ℒh = {"HW: Working", "HW: Failure"});
PlotDiagram[ℳh, 𝒲h, ℒh]
```

Out[ ]=



b) The probability expressions for a two state Markov model were derived in Lab III. Hence, we have that P_0 is given by:

$$P_0 = \frac{\lambda_h}{\lambda_h + \mu_h}$$

In[ ]:=
```
𝒫h = {λh → 1 / 14, μh → 1};
p_unavailable = λh / (λh + μh) /. 𝒫h
```

Out[ ]=

$$\frac{1}{15}$$

The probability P_0 is calculated to be $\frac{1}{15} \approx 0.0667$, which means that a bus is unavailable 6.67% of the time.
We can also check this using Mathematica features

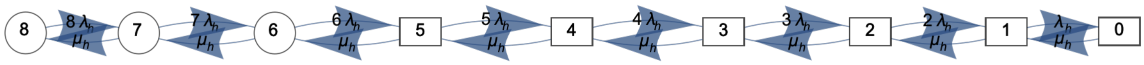
In[ ]:=
```
ps = ProbStationary[ℳh /. 𝒫h ];
Print["Unavailability, U = ", U = UnAvailability[ps, 𝒲h]]
```

Unavailability, U = $\frac{1}{15}$

c) In this model we have one state variable describing how many of the buses are operational. The events are defined by λ_h and μ_h, where λ_h is the rate of hardware failure, and μ_h is the repair rate. Since $\lambda_h$is the rate per bus, the rate is dependent of number of operational buses. Even though the system will work up until there is no operational buses, I have chosen to define the working states to only be then we have 6 or more working buses, because that is what we want in order to serve the demand without too much delays

In[ ]:=
```
ClearAll[Q];

Q = Table[0, {9}, {9}];
For[k = 1, k ≤ 8, k++,
   Q⟦k, k + 1⟧ = μh;
   Q⟦k + 1, k⟧ = k * λh;
  ];

ℳc = SetDiagonal[Transpose[Q]];
𝒲c = {False, False, False, False, False, False, True, True, True};
ℒc = Range[0, 8];

PlotDiagram[ℳc, 𝒲c, ℒc]
```

Out[ ]=



d) The steady state probabilities show how the number of repairmen affects the system, by influencing the likelihood of having more operational buses. With one repairman, $s = 1$, the system often has 8 operating buses, but there is still a chance of having less than 8, and in our case - less than 6. When adding a second repairman, $s = 2$, this instantly increases the probability for the higher operational states. We can also see that adding a third repairman, has a rather small impact on the system, which indicates that $s = 2$ makes the ideal number of repairmen. Index one in the list shows the state probability for one operational bus, etc.

In[ ]:=
```
ClearAll[Q, s];

Q = Table[0, {9}, {9}];
For[k = 1, k ≤ 8, k++,
   Q⟦k, k + 1⟧ = μh;
   Q⟦k + 1, k⟧ = k * λh;
  ];
Q⟦8, 9⟧ = Min[1, μh];
Q⟦7, 8⟧ = Min[2, μh];

ℳc = SetDiagonal[Transpose[Q]];
𝒲c = {False, False, False, False, False, False, True, True, True};

results = {};
Do[
 ps = ProbStationary[ℳc /. {λh → (λh /. 𝒫h), μh → (μh /. 𝒫h) * s}] // N;
 AppendTo[results, ps];
 Print["Steady state probabilities for s = ", s, ": ", ps] , {s, 1, 3}
]
```

```
Steady state probabilities for s = 1: {0.0000133998, 0.000187598,
  0.00131318, 0.00612819, 0.0214487, 0.0600562, 0.140131, 0.280262, 0.490459}
Steady state probabilities for s = 2: {1.21883 × 10^-7, 3.41271 × 10^-6,
  0.000047778, 0.000445928, 0.0031215, 0.0174804, 0.0815751, 0.3263, 0.571026}
Steady state probabilities for s = 3: {1.07856 × 10^-8, 4.52997 × 10^-7,
  9.51293 × 10^-6, 0.000133181, 0.0013984, 0.0117466, 0.082226, 0.328904, 0.575582}
```

e) Mean time to first failure is the average time of functionality before a failure impacts the product/system for the first time.

```
In[ ]:= Do[Print["MTFF for s = ", s, ": ",
  MTFF[ℳc /. {λh → (λh /. 𝒫h), μh → (μh /. 𝒫h) * s}, Reverse[𝒲c]] // N], {s,
  1, 3}]
```

```
MTFF for s = 1: 3.22449
MTFF for s = 2: 1.55485
MTFF for s = 3: 1.02419
```

f) Mean time to failure is the average time a product/system operates before a failure a occurs.

```
In[ ]:= Do[Print["MTTF for s = ", s, ": ",
  MTTF[ℳc /. {λh → (λh /. 𝒫h), μh → (μh /. 𝒫h) * s}, 𝒲c ] // N], {s, 1, 3}]
```

```
MTTF for s = 1: 20.7628
MTTF for s = 2: 34.0625
MTTF for s = 3: 34.0625
```

g) Mean down time is a measure for the average time a product/system is down after failure while it is being repaired and maintained. In other words, the average time it is non-operational.

```
In[ ]:= Do[Print["MDT for s = ", s, ": ",
  MDT[ℳc /. {λh → (λh /. 𝒫h), μh → (μh /. 𝒫h) * s}, 𝒲c ] // N], {s, 1, 3}]
```

```
MDT for s = 1: 1.4844
MDT for s = 2: 0.603509
MDT for s = 3: 0.377078
```

h) Unavailability measures the proportion of time that a system is non-operational. We can use this proportion to calculate the total of downtime in our system, in this case, on an annual basis.

```
In[ ]:= unavailabilities = {};

Do[
 unavailable = UnAvailability[results⟦s⟧, 𝒲c] // N;
 AppendTo[unavailabilities, unavailable];
 Print["Unavailablity for s = ", s, ": ", unavailabilities⟦s⟧], {s, 1, 3}
]

hoursYear = 365 * 24;

Do[
 Print["Expected hours of downtime per year for s = ",
  s, ": ", unavailabilities⟦s⟧ * hoursYear], {s, 1, 3}
]
```

```
Unavailablity for s = 1: 0.0891472
Unavailablity for s = 2: 0.0210991
Unavailablity for s = 3: 0.0132881
Expected hours of downtime per year for s = 1: 780.93
Expected hours of downtime per year for s = 2: 184.828
Expected hours of downtime per year for s = 3: 116.404
```

i) After analyzing the different metrics, we observe that they all improve as the number of repairmen increase, with the most significant improvement occurring when the number of repairmen increase from one to two. This change leads to a significant reduction in downtime and enhances system availability. Adding a third repairman continues to improve the metrics, even though the improvement of the metrics are remarkably smaller, indicating that this addition is rather unnecessary.

Task IV.A.2

Now, consider only software failures. There are no constraints regarding repairmen because it is assumed that the software failures are repaired by rebooting the software on the bus (at an end station).

Two types of software failures are included:
1. *Omission failures* : the failure is detected and the bus has to return to the depot and be diagnosed and restarted (autonomous process), with failure rate $T_o \sim n.e.d(\lambda_o)$ and repair rate $T_{ro} \sim n.e.d(\mu_o)$. The bus cannot serve any passengers in this state.
2. *Value failures* : the information sent and received is valid but wrong (undetected), with failure rate $T_v \sim n.e.d(\lambda_v)$. Value failures are not detected and repaired and will be in the system until you reboot the bus. A reboot is only done after an *omission* failure has been detected. The omission failure rate after a value failure is doubled, i.e., $2\,\lambda_o$. The bus is not able to select the best route

and therefor only 60% of the passengers are served (within acceptable waiting time).

a) Define the state variable and events, and draw the Markov model.
b) Use Mathematica to obtain the *steady state probabilities*. Assume λ_o = 1/2 [once every second day], and μ_o = 8 [repairs per day], and λ_v = 1/4 [once every forth day],

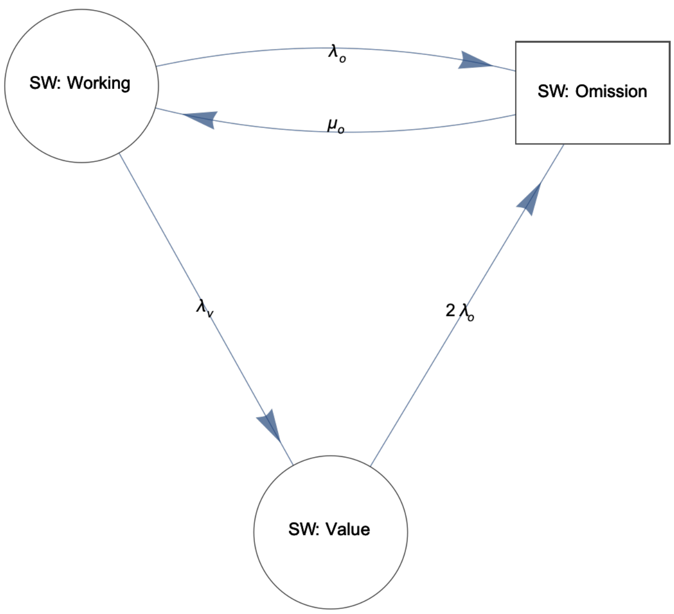
```
𝒫s = {λo → 1 / 2, μo → 8, λv → 1 / 4}.
```

c) Define the down states in this model and obtain the *availability*.
d) What is the average % of passengers the bus can handle? Discuss this relative to availability.

a) I have defined one state variable, that can take one of three different values: {“Working”, “Omission”, and “Value”}. The event are defined by λ_o, λ_v, and μ_o, where λ_o represents the rate of omission failure, λ_v represents the rate of value failure, and μ_o represents the repair rate after omission failure.

In[◦]:=
```
ClearAll[Q]
Q = Table[0, {3}, {3}];
Q⟦1, 2⟧ = λo;
Q⟦2, 1⟧ = μo;
Q⟦1, 3⟧ = λv;
Q⟦3, 2⟧ = 2 * λo;

ℳs = SetDiagonal[Transpose[Q ]];
𝒲s = {True, False, True};
ℒs = {"SW: Working", "SW: Omission", "SW: Value"};

PlotDiagram[ℳs, 𝒲s, ℒs]
```

Out[◦]=



b) The steady state probabilities of the system is calculated below.

```
In[∘]:= 𝒫s = {λo → 1 / 2, μo → 8, λv → 1 / 4};
ps = ProbStationary[ℳs /. 𝒫s] // N;
Print["Steady state probabilities: ", ps]

Steady state probabilities: {0.744186, 0.0697674, 0.186047}
```

c) The only down state in this model is when a omission failure occurs. The availability will be the amount of time the bus is not in this down state.

```
In[∘]:= sAvailability = 1 - UnAvailability[ps, 𝒲s];
Print["The availability of the bus is ", sAvailability]

The availability of the bus is 0.930233
```

d) When a value fail has occurred the bus will still be available, but a consequence of this error is that the bus is not able to select the best route and only serves 60% of the passengers. Because of this, I have calculated the average passenger handling with respect to this limitation.

```
In[∘]:= sAverage = ps[[1]] + ps[[2]] * 0.6;
Print["On average, the bus can handle ", sAverage * 100, "% of passengers."]

On average, the bus can handle 78.6047% of passengers.
```

Task IV.A.3*

A bus can only drive if both the hardware and the software of the bus is working. The objective is still to obtain the steady state probability of the number of failed busses in the system.

a) Combine the models $\mathcal{M}_h$ and $\mathcal{M}_s$ from above into a new model $\mathcal{M}_c$ [use Köeneker-sum]. Plot the diagram of $\mathcal{M}_c$.
b) Obtain the availability and average number of passengers handled by a bus.
The bus that gets a mechanical failure will be returned to the garage for repair. Any software failure (omission or value) will also be removed when the bus is repaired.
c) Modify the transitions in model $\mathcal{M}_c$ to include the fact that software failures are also removed after a repair. Show this on paper.
d) Implement the changes $\mathcal{M}_c$ by use of the ChangeTransition[] function provided below. Call this new model $\mathcal{M}_x$, and plot the diagram of $\mathcal{M}_x$.
e) Obtain the availability and average number of passengers handled by a bus in model $\mathcal{M}_x$ and compare it with the values from IV.A.3.b) for $\mathcal{M}_c$. Based on this, would you say that the $\mathcal{M}_c$ is a good approximation of $\mathcal{M}_x$?
In principle, you can combine the bus model to form a fleet of busses (e.g., 8 as above).
f) If you have 8 busses, and each bus is modelled by $\mathcal{M}_c$ (or $\mathcal{M}_x$): how many states will you have when you combine the models by Kronecker sum?
g) Combine 2 (only!) busses into a model $\mathcal{M}_t$, and plot the diagram for this model, use KroenSum[] again. Comment on how useful this model is? (how readable it is, what metrics can you obtain). How many states do you have in the model?
h) To make the model more scalable, how can you model and obtain the (un)availability if you

ignore the fact that you have a limited number of repairmen. Assume that both busses needs to be operational (“Up”) in order for the bus system to be in operation (the system is “Up”). Compare the system unavailability from the combined model $\mathcal{M}_t$ with the alternative approach you suggest.

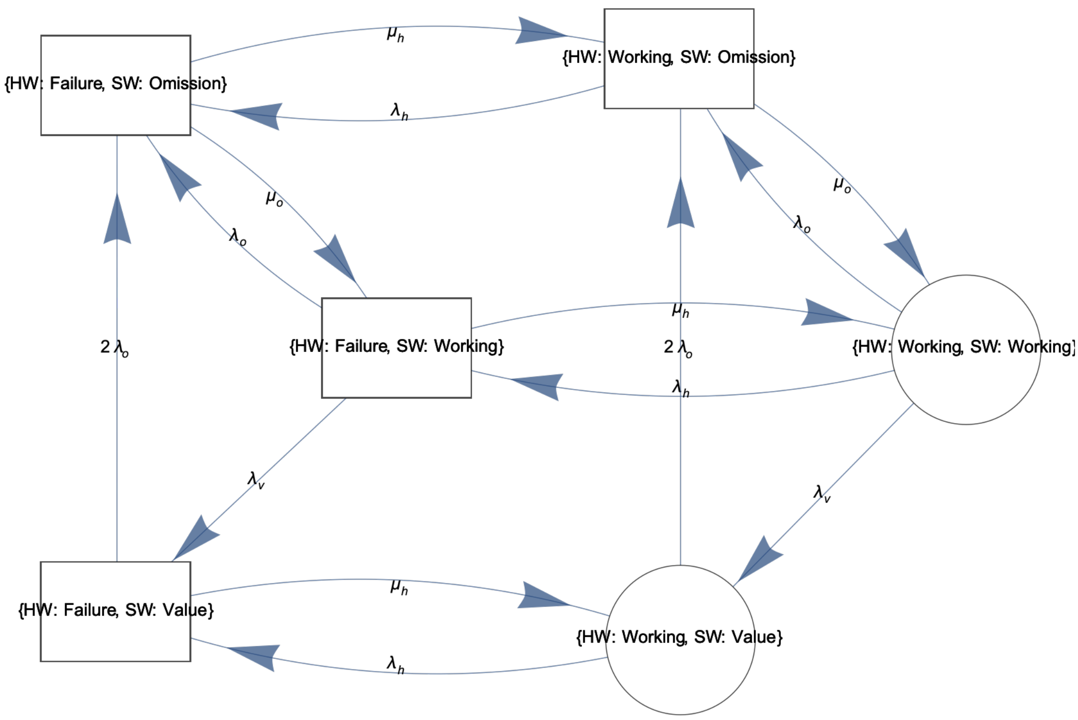
Module for adding and deleting transitions in $\mathcal{M}_T$

In[ ]:=
```
(* Function that adds transitions with rate from state a→
 b in model M (state indexes in L) *)
(* Transitions can be deleted by letting rate=0 *)
(* Note: M is transposed such that the cell (b,a) is from a to b *)
(* Note:
 SetDiagonal is needed to change the diagonal vector of M appropriatly *)
ChangeTransition[M_, L_, a_, b_, rate_] :=
  Module[{tmp, idxa, idxb},
   idxa = Flatten[Position[# == a & /@ L, True]];
   idxb = Flatten[Position[# == b & /@ L , True]];
   tmp = M;
   tmp[[idxb, idxa]] = rate;
   SetDiagonal[tmp]
   ]
```

a) Using the Köeneker-sum, $\mathcal{M}_c$ has been plotted below. We can see that we have a new diagram combining all states from our previous models $\mathcal{M}_\mathrm{h}$ and $\mathcal{M}_\mathrm{s}$.

In[ ]:=
```
ℳc = KroenSum[ℳh, ℳs];
ℒc = LabelList[ℒh, ℒs];
𝒲c = SeriesMode[𝒲h, 𝒲s];
PlotDiagram[ℳc, 𝒲c, ℒc]
```

Out[ ]=

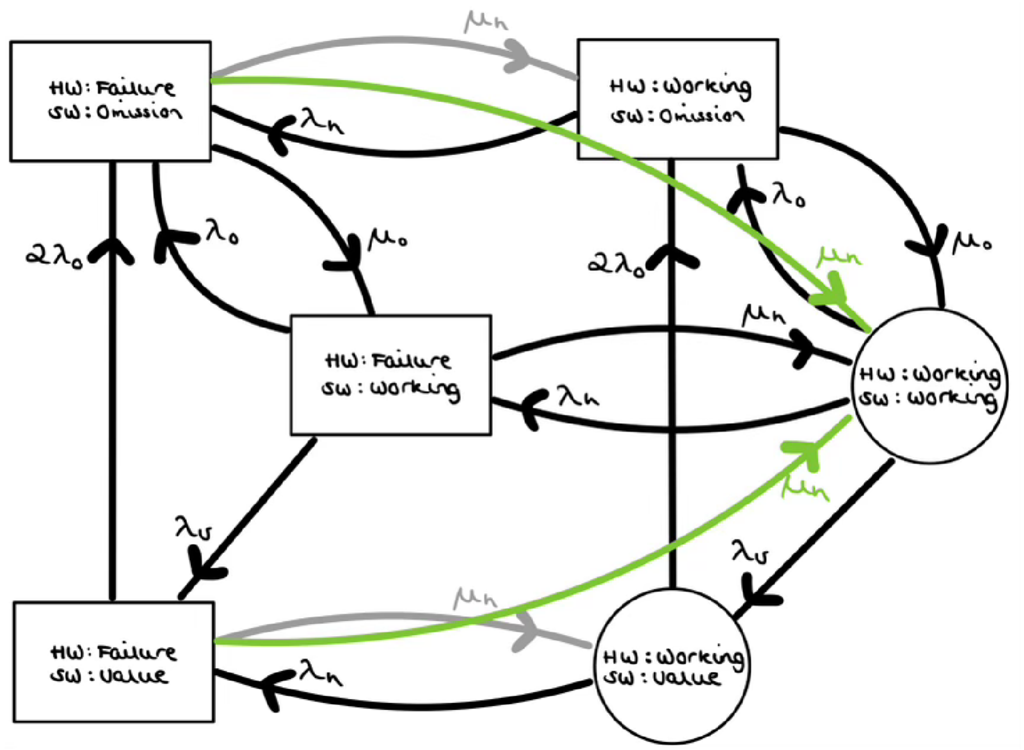


b) Since we have no way to calculate the number of passengers, the percentage of passenger handling is calculated instead. This is done in the same way as in IV.A.2.

In[ ]:=
```
Clear[ps]
ps = ProbStationary[ℳc /. 𝒫s /. 𝒫h ] // N;
Print["Steady state probabilities: ", ps]
cAvailability = 1 - UnAvailability[ps, 𝒲c];
Print["The availability of the bus is ", cAvailability]
cAverage = ps[[1]] + ps[[3]] * 0.6;
Print["On average, the bus can handle ",
 cAverage * 100, "% of the passengers."]
```

```
Steady state probabilities:
 {0.694574, 0.0651163, 0.173643, 0.0496124, 0.00465116, 0.0124031}
The availability of the bus is 0.868217
On average, the bus can handle 79.876% of the passengers.
```

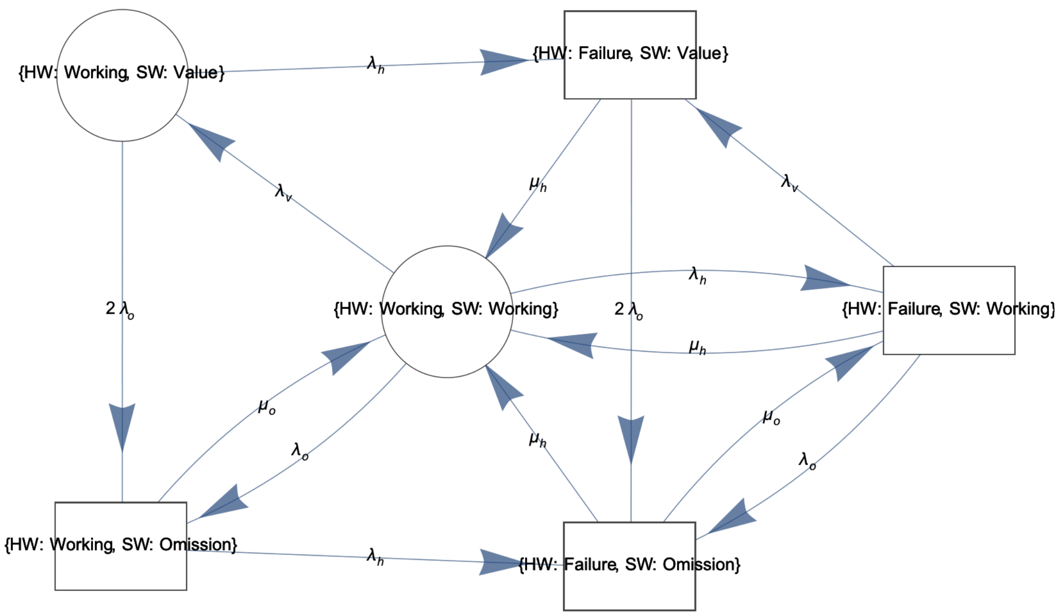
c) The new Markov model is illustrated below. The grey transitions are the ones we wanted to remove, and the green represents the new transitions.



d) Using ChangeTransition to add and remove transitions. New model is plotted below.

```
In[ ]:= 𝓜x = 𝓜c;
𝓛x = 𝓛c;
𝓦x = 𝓦c;

(*Removing the grey transitions*)
𝓜x = ChangeTransition[𝓜x, 𝓛x,
   {"HW: Failure", "SW: Omission"}, {"HW: Working", "SW: Omission"} , 0];
𝓜x = ChangeTransition[𝓜x, 𝓛x,
   {"HW: Failure", "SW: Value"}, {"HW: Working", "SW: Value"} , 0];

(*Adding the green transitions*)
𝓜x = ChangeTransition[𝓜x, 𝓛x,
   {"HW: Failure", "SW: Omission"}, {"HW: Working", "SW: Working"} , μh];
𝓜x = ChangeTransition[𝓜x, 𝓛x,
   {"HW: Failure", "SW: Value"}, {"HW: Working", "SW: Working"} , μh];

PlotDiagram[𝓜x, 𝓦x, 𝓛x]
```

Out[ ]=



e) Since all of the measurements are nearly identical for the two models, we can say that $\mathcal{M}_c$ is a good approximation of $\mathcal{M}_x$.

In[ ]:=
```
𝒫x = {λh → 1 / 14, μh → 1, λo → 1 / 2, μo → 8, λv → 1 / 4};
ps = ProbStationary[ℳx /. 𝒫x] // N;
Print["Steady state probabilities: ", ps]
xAvailability = 1 - UnAvailability[ps, 𝒲x];
Print["The availability of the bus is ", xAvailability]
xAverage = ps⟦1⟧ + ps⟦3⟧ * 0.6;
Print["On average, the bus can handle ",
 xAverage * 100, "% of the passengers."]
```

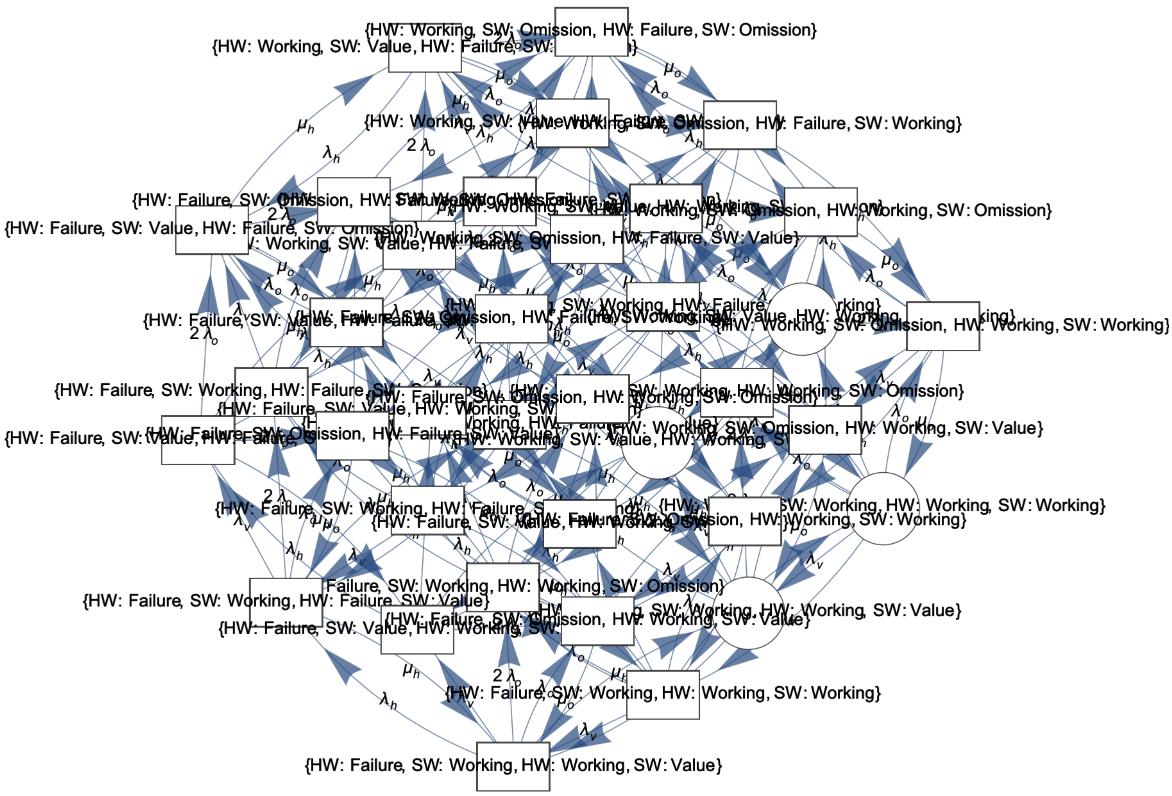
```
Steady state probabilities:
 {0.704834, 0.064038, 0.164461, 0.0499246, 0.00462785, 0.0121142}
The availability of the bus is 0.869295
On average, the bus can handle 80.3511% of the passengers.
```

f) When each bus has 6 states, you will get 6^n states when you combine n buses. If $n = 8$, we have $6^8 = 1\ 679\ 616$ states in total.

g) This model has $6^2 = 36$ states. As you can see, the model is very hard to read and not very helpful. I could have changed the labels somehow, but I think that would have made a small difference to the model overall. Using Mathematica , we can obtain the same metrics as before, but it is very hard to get any information out of the model itself.

In[•]:=
```
𝓜t = KroenSum[𝓜h, 𝓜s, 𝓜h, 𝓜s];
ℒt = LabelList[ℒh, ℒs, ℒh, ℒs];
𝒲t = SeriesMode[𝒲h, 𝒲s, 𝒲h, 𝒲s];
PlotDiagram[𝓜t, 𝒲t, ℒt]
```

Out[•]=



h) My suggestion for an alternative approach to calculate the availability of several buses is the following:

In[•]:=
```
ps = ProbStationary[𝓜t /. 𝒫x] // N;
Print["The unavailability of the buses in the extended model is: ",
 1 - UnAvailability[ps, 𝒲t]]
Print["The unavailabilty of one bus combineds as if there were two: ",
 cAvailability^2 ]
```

```
The unavailability of the buses in the extended model is: 0.753801

The unavailabilty of one bus combineds as if there were two: 0.753801

The unavailabilty of one bus combineds as if there were two: 0.753801
```

The first method uses a the model to determine availability, while the second method assumes the buses operate independently, allowing us to approximate the availability by squaring the availability of a single bus. This second method is simpler, faster, and gives, in this case, the exact same result as the complex model, making it a more practical alternative. When we have a large amount of buses, the number of states to investigate will get very big. Therefore, combining the results for one bus gives a simpler calculation, but still a reasonable value.

Task IV.B - Road blocks after severe weather

In this task you are going to study the effect of severe weather that leads to road blocks. Assume now that the busses can take any of the available routes (no preplanned routes as in Lab III). The time between severe weather is T_{wo} and it last T_{wl}. Both are n.e.d., i.e., $T_{wo} \sim n.e.d(\alpha)$ and $T_{wl} \sim n.e.d(\beta)$. Numerical values are: $\mathcal{P}_r = \{\alpha \to 1/30, \beta \to 1/2\}$ [unit is 1/day].

a) What is the availability and reliability of road *i* if the weather "hits" this road with the probability p_i? (Symbolic expressions expected)
b) What is the availability and reliability after t=100 days of the different roads if the weather "hits" a road *i* with the probabilities
$\mathcal{P}_a = \{p_1 \to 0.0714, p_2 \to 0.0714, p_3 \to 0.0714, p_4 \to 0.1, p_5 \to 0.0714, p_6 \to 0.1, p_7 \to 0.1, p_8 \to 0.0483,$
$p_9 \to 0.0483, p_{10} \to 0.0483, p_{11} \to 0.0483, p_{12} \to 0.0714, p_{13} \to 0.0714, p_{14} \to 0, p_{15} \to 0.0714\}$
c) Which assumptions do you have to make in order to use Reliability Block Diagram (RBD)?
d) Assume that road R_{14} in the system description is under maintenance and therefor always blocked. Make Reliability Block Diagram (RBD) for the following end stop combinations [one for each]

- $\mathcal{M}_{RBD1}$, the bus routes between $ES_1 \to ES_3$(only eastbound)
- $\mathcal{M}_{RBD2}$, the bus routes between $ES_1 \to ES_4$(only eastbound)
- $\mathcal{M}_{RBD3}$, the bus routes between $ES_2 \to ES_3$(only eastbound)
- $\mathcal{M}_{RBD4}$, the bus routes between $ES_2 \to ES_4$(only eastbound)

e) Use your models to obtain the availability (both symbolically and numerically) of the four alternative end stop, ES_i combinations. Which of the four has highest availability? What is the probability that bus stop 5 cannot be served (is unavailable)?
f) Obtain the reliability function, $R_{13}(t)$, of $ES_1 \to ES_3$. Plot the function with parameters in $\mathcal{P}_r$ and $\mathcal{P}_a$. What is the probability that the bus service between $ES_1 \to ES_3$ works without interruptions for 10 days?
g*) Assume that R_{13} is never blocked by sever weather. What is the most dominant/likely (combination of) roads wrt. probability of the routes between $ES_1 \to ES_3$? [hint: define (minimum) cut sets].

a) The availability for the weather to hit the road is given by

$$\mathcal{A} = \frac{\beta}{\alpha + \beta}$$

In our case, the road is split into several roads. This gives the availability of road i as

$$\mathcal{A}_i = \frac{\beta}{p_i\, \alpha + \beta}$$

The reliability of road i at time t is given by

$$\mathcal{R}_i(p, t) = e^{-p_i\, \alpha\, t}$$

In[ ]:=
```
roadAvailability[p_] := β / (p * α + β);
roadReliability[t_, p_] := Exp[-p * α * t];
```

b) The availability and reliability after t = 100 days of the different roads are represented in the table below.

```
In[ ]:= α = 1 / 30;
β = 1 / 2;
𝒫a = {0.0714, 0.0714, 0.0714, 0.1, 0.0714, 0.1, 0.1,
   0.0483, 0.0483, 0.0483, 0.0483, 0.0714, 0.0714, 0, 0.0714};

𝒜 = {a1 → roadAvailability[𝒫a⟦1⟧], a2 → roadAvailability[𝒫a⟦2⟧],
   a3 → roadAvailability[𝒫a⟦3⟧], a4 → roadAvailability[𝒫a⟦4⟧],
   a5 → roadAvailability[𝒫a⟦5⟧], a6 → roadAvailability[𝒫a⟦6⟧],
   a7 → roadAvailability[𝒫a⟦7⟧], a8 → roadAvailability[𝒫a⟦8⟧],
   a9 → roadAvailability[𝒫a⟦9⟧], a10 → roadAvailability[𝒫a⟦10⟧],
   a11 → roadAvailability[𝒫a⟦11⟧], a12 → roadAvailability[𝒫a⟦12⟧],
   a13 → roadAvailability[𝒫a⟦13⟧], a14 → roadAvailability[𝒫a⟦14⟧],
   a15 → roadAvailability[𝒫a⟦15⟧]};

ℛ[t_] = {r1 → roadReliability[t, 𝒫a⟦1⟧], r2 → roadReliability[t, 𝒫a⟦2⟧],
    r3 → roadReliability[t, 𝒫a⟦3⟧], r4 → roadReliability[t, 𝒫a⟦4⟧],
    r5 → roadReliability[t, 𝒫a⟦5⟧], r6 → roadReliability[t, 𝒫a⟦6⟧],
   r7 → roadReliability[t, 𝒫a⟦7⟧], r8 → roadReliability[t, 𝒫a⟦8⟧],
   r9 → roadReliability[t, 𝒫a⟦9⟧], r10 → roadReliability[t, 𝒫a⟦10⟧],
   r11 → roadReliability[t, 𝒫a⟦11⟧], r12 → roadReliability[t, 𝒫a⟦12⟧],
   r13 → roadReliability[t, 𝒫a⟦13⟧], r14 → roadReliability[t, 𝒫a⟦14⟧],
   r15 → roadReliability[t, 𝒫a⟦15⟧]};

ℛ100 = Block[{t = 100}, ℛ[t]];

results = TableForm[
   Transpose[{𝒫a, 𝒜, ℛ100}],
   TableHeadings →
    {None, {"Probability", "Availability", "Reliability after 100 days"}}];
results
```

Out[∘]//TableForm=

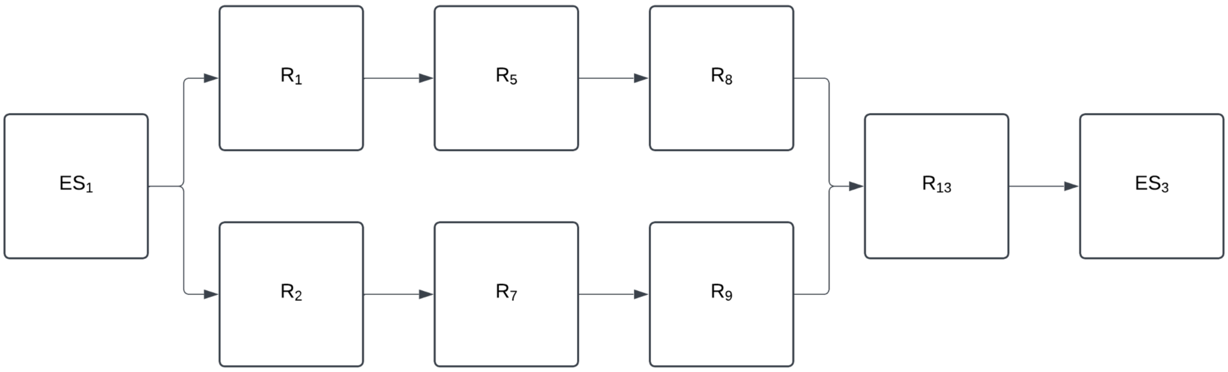
Probability	Availability	Reliability after 100 days
0.0714	$a_1 \to 0.995263$	$r_1 \to 0.788203$
0.0714	$a_2 \to 0.995263$	$r_2 \to 0.788203$
0.0714	$a_3 \to 0.995263$	$r_3 \to 0.788203$
0.1	$a_4 \to 0.993377$	$r_4 \to 0.716531$
0.0714	$a_5 \to 0.995263$	$r_5 \to 0.788203$
0.1	$a_6 \to 0.993377$	$r_6 \to 0.716531$
0.1	$a_7 \to 0.993377$	$r_7 \to 0.716531$
0.0483	$a_8 \to 0.99679$	$r_8 \to 0.851292$
0.0483	$a_9 \to 0.99679$	$r_9 \to 0.851292$
0.0483	$a_{10} \to 0.99679$	$r_{10} \to 0.851292$
0.0483	$a_{11} \to 0.99679$	$r_{11} \to 0.851292$
0.0714	$a_{12} \to 0.995263$	$r_{12} \to 0.788203$
0.0714	$a_{13} \to 0.995263$	$r_{13} \to 0.788203$
0	$a_{14} \to 1$	$r_{14} \to 1$
0.0714	$a_{15} \to 0.995263$	$r_{15} \to 0.788203$

c) In order to make a Reliability Block Diagram we need to assume the following:

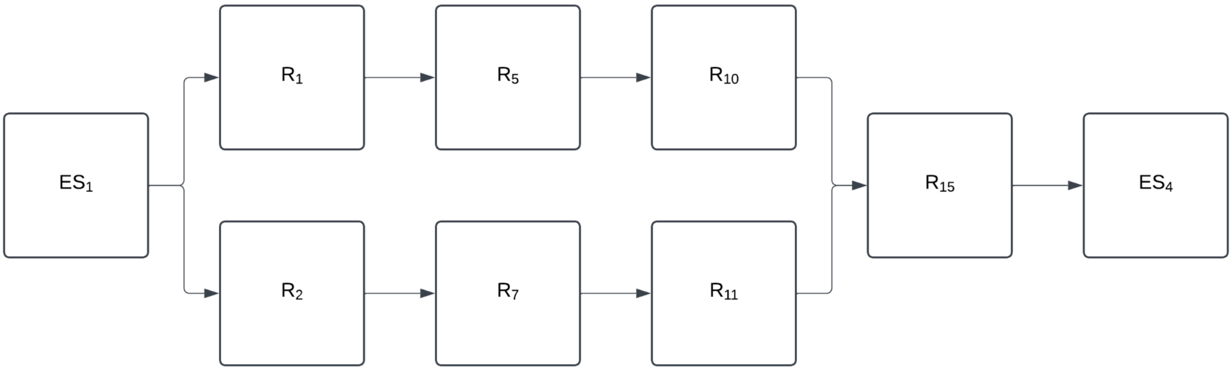
- All systems fail independent of other subsystems, given that the system consists of more than one subsystem.
- All systems are restored independent of the other subsystems, after failure
- The system works as it is supposed to regarding repairs and

d)

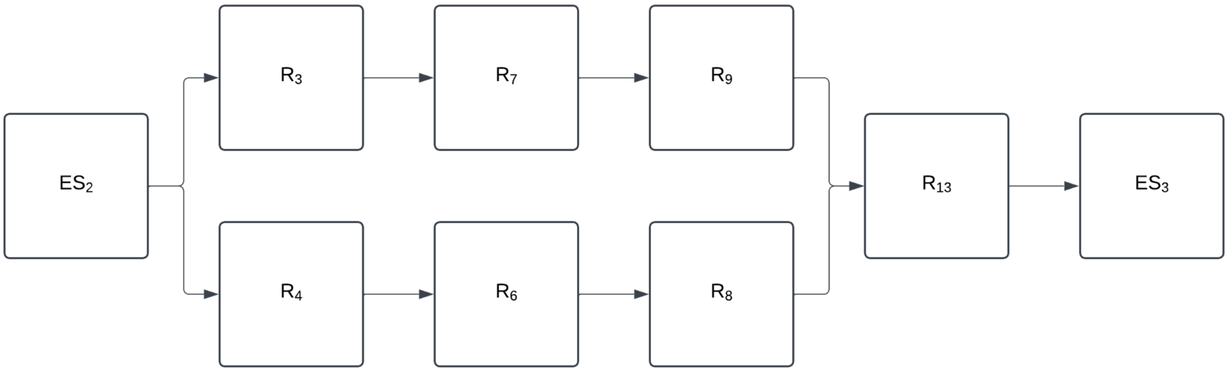
- $\mathcal{M}_{\text{RBD1}}$, the bus routes between $ES_1 \to ES_3$(only eastbound)



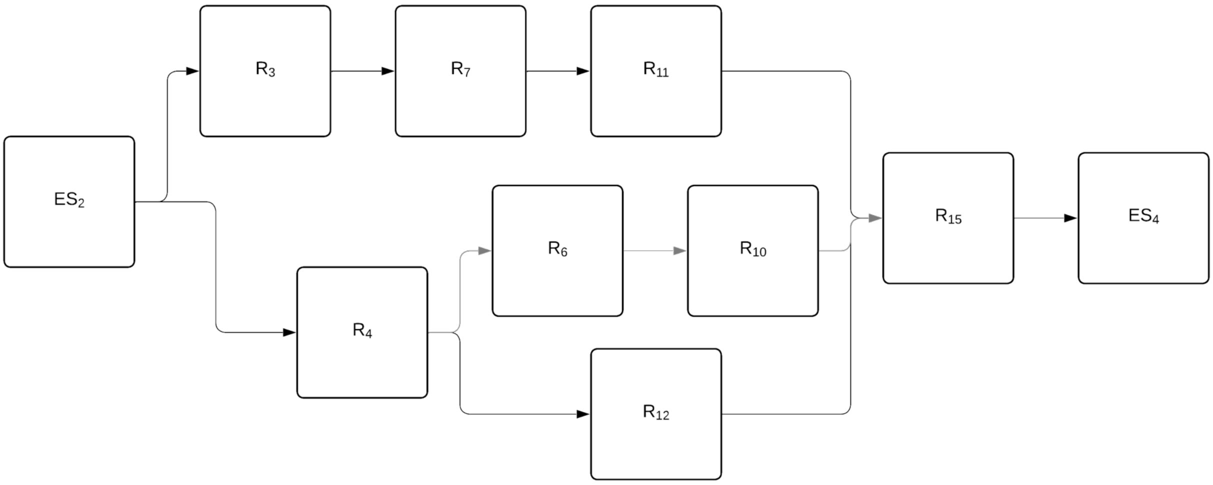
- $\mathcal{M}_{\text{RBD2}}$, the bus routes between $ES_1 \to ES_4$(only eastbound)



- $\mathcal{M}_{\text{RBD3}}$, the bus routes between ES$_2$ → ES$_3$(only eastbound)



- $\mathcal{M}_{\text{RBD4}}$, the bus routes between ES$_2$ → ES$_4$(only eastbound)



e) The availability of a system in series is given by

$$\mathcal{A}_{\text{series}} = \Pi_{i=1}^{n} A_i$$

and in parallel

$$\mathcal{A}_{\text{parallel}} = 1 - \Pi_{i=1}^{n} (1 - A_i)$$

The formulas for our end stop combinations are derived from the equations above. They can be observed below.

```
In[∘]:= (*Availability from end stop to end stop*)
𝒜RBD1 = (1 - (1 - (a1 * a5 * a8)) * (1 - (a2 * a7 * a9))) * a13;
𝒜RBD2 = (1 - (1 - (a1 * a5 * a10)) * (1 - (a2 * a7 * a11))) * a15;
𝒜RBD3 = (1 - (1 - (a3 * a7 * a9)) * (1 - (a4 * a6 * a8))) * a13;
𝒜RBD4 = (1 - (1 - a3 * a7 * a11) * (1 - a4 (1 - (1 - a6 * a10) (1 - a12)))) * a15;

Print["ES1->ES3: Availability 𝒜 = ", 𝒜RBD1, " = ", 𝒜RBD1 /. 𝒜]
Print["ES1->ES4: Availability 𝒜 = ", 𝒜RBD2, " = ", 𝒜RBD2 /. 𝒜]
Print["ES2->ES3: Availability 𝒜 = ", 𝒜RBD3, " = ", 𝒜RBD3 /. 𝒜]
Print["ES2->ES4: Availability 𝒜 = ", 𝒜RBD4, " = ", 𝒜RBD4 /. 𝒜]
```

ES_1->ES_3: Availability $\mathcal{A}$ = $(1-(1-a_1\,a_5\,a_8)\,(1-a_2\,a_7\,a_9))\,a_{13}$ = 0.99508

ES_1->ES_4: Availability $\mathcal{A}$ = $(1-(1-a_1\,a_5\,a_{10})\,(1-a_2\,a_7\,a_{11}))\,a_{15}$ = 0.99508

ES_2->ES_3: Availability $\mathcal{A}$ = $(1-(1-a_4\,a_6\,a_8)\,(1-a_3\,a_7\,a_9))\,a_{13}$ = 0.995026

ES_2->ES_4: Availability $\mathcal{A}$ = $(1-(1-a_3\,a_7\,a_{11})\,(1-a_4\,(1-(1-a_6\,a_{10})\,(1-a_{12}))))\,a_{15}$ = 0.995166

We can see that $ES_2 \rightarrow ES_4$ has the highest availability, although the margin is quite small.

When calculating the probability that bus stop 5 is unavailable, we need to check the probability that R_7, or R_2 and $R_3$are unavailable, as those roads are the ones leading to S_5. Making an expression for $\mathcal{S}_5$ unavailability using the inclusion-exclusion principle:

```
In[∘]:= unavailr2 = 1 - a2 /. 𝒜;
unavailr3 = 1 - a3 /. 𝒜;
unavailr7 = 1 - a7 /. 𝒜;

unavail𝒮5 = unavailr7 + unavailr2 * unavailr3 - unavailr7 * unavailr2 * unavailr3;
Print["The probability of stop S5 being unavailable is ", unavail𝒮5 "."]
```

The probability of stop S_5 being unavailable is 0.00664481 .

f) The reliability of a system in series is given by

$$\mathcal{R}_{\text{series}}(t) = \prod_{i=1}^{n} R_i(t)$$

and in parallel

$$\mathcal{R}_{\text{parallel}}(t) = 1 - \prod_{i=1}^{n} (1 - R_i(t))$$

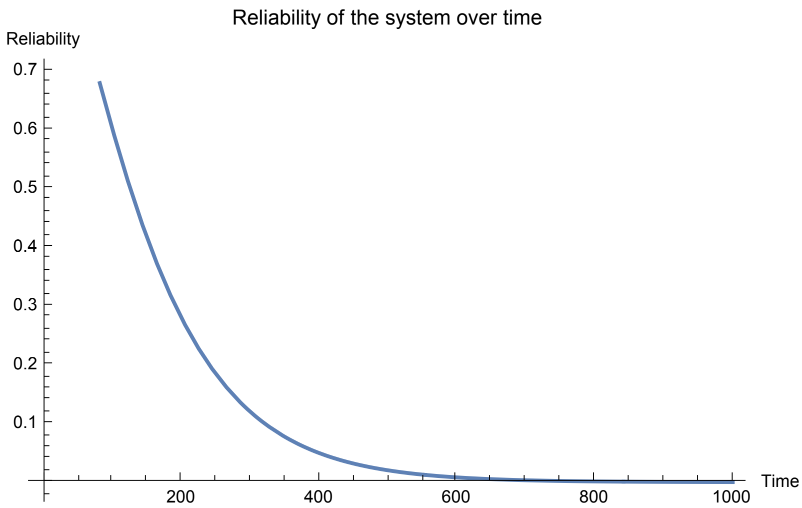
Thus, we get that the reliability function for $ES_1 \rightarrow ES_3$ is

$$\mathcal{R}_{13}(t) = (1 - (1 - (r_1(t)\; r_5(t)\; r_8(t)))(1 - (r_2(t)\; r_7(t)\; r_9(t))))\; r_{13}(t)$$

In[∘]:=
```
𝓡10 days = (1 - (1 - (r1 * r5 * r8)) * (1 - (r2 * r7 * r9))) * r13 /. 𝓡[10];
Print["The probability that the bus service between ES1 -> ES3
   works without interruptions for 10 days is ", 𝓡10 days , "."]

𝓡13 = (1 - (1 - (r1 * r5 * r8)) * (1 - (r2 * r7 * r9))) * r13 ;

Plot[𝓡13 /. 𝓡[t], {t, 0, 1000}, AxesLabel → {"Time", "Reliability"},
 PlotLabel → "Reliability of the system over time" ]
```

```
The probability that the bus service between
   ES1 -> ES3 works without interruptions for 10 days is 0.972226.
```

Out[∘]=



Here we can see that the reliability function behaves as a n.e.d for big values of t.

g) To make our calculations simpler, we define cut sets to identify the minimal combinations of component failures that will lead to a system failure. The minimal cut sets of $ES_1 \to ES_3$ are:

$set_1 = \{R_1, R_2\}$
$set_2 = \{R_1, R_7\}$
$set_3 = \{R_1, R_9\}$
$set_4 = \{R_5, R_2\}$
$set_5 = \{R_5, R_7\}$
$set_6 = \{R_5, R_9\}$
$set_7 = \{R_8, R_2\}$
$set_8 = \{R_8, R_7\}$
$set_9 = \{R_8, R_9\}$

```
In[ ]:= sets = {};
firstIndex = {1, 5, 8};
secondIndex = {2, 7, 9};

Do[
    set = 𝒫a⟦firstIndex⟦i⟧⟧ * 𝒫a⟦secondIndex⟦j⟧⟧;
    AppendTo[sets, set];,
    {i, Length[firstIndex]}, {j, Length[secondIndex]}
]

Print["Probability of the cut sets are: ", sets]
maxValue = Max[sets];
maxIndices = Flatten[Position[sets, maxValue]];
Print["Indices of the maximum cut sets: ", maxIndices];

Probability of the cut sets are: {0.00509796, 0.00714, 0.00344862,
  0.00509796, 0.00714, 0.00344862, 0.00344862, 0.00483, 0.00233289}

Indices of the maximum cut sets: {2, 5}
```

This indicates that $\text{set}_2 = \{R_1,\ R_7\}$ and $\text{set}_5 = \{R_5,\ R_7\}$ are the most likely to be unavailable due to severe weather. The critical roads identified are R_1, R_5, and R_7, suggesting that these roads require the highest priority for weather-related precautions.

Task IV.C - DDOS attack on the bus coordination system

To select the best route (according to the policy you have defined), you need correct and updated information about the state (e.g., how many passengers waiting, how long they have waited) at each bus stop. In this task, you are going to study the effect of lost informations after a DDOS attack on the bus company's server that hosts the bus coordination system. The busses will still run, but have to select routes blindly (e.g., randomly) without any knowledge about where the passengers are.

IV.C.1

First, we will make a change in the bus system. We will consider only busses from station ES_1 (east-bound routes) to three end stations (denoted ES_2, ES_3, ES_4). We now have 10 stops (denoted by the number 1-10), and 21 roads (defined as connecting bus stops i and j: {i⟶j}).

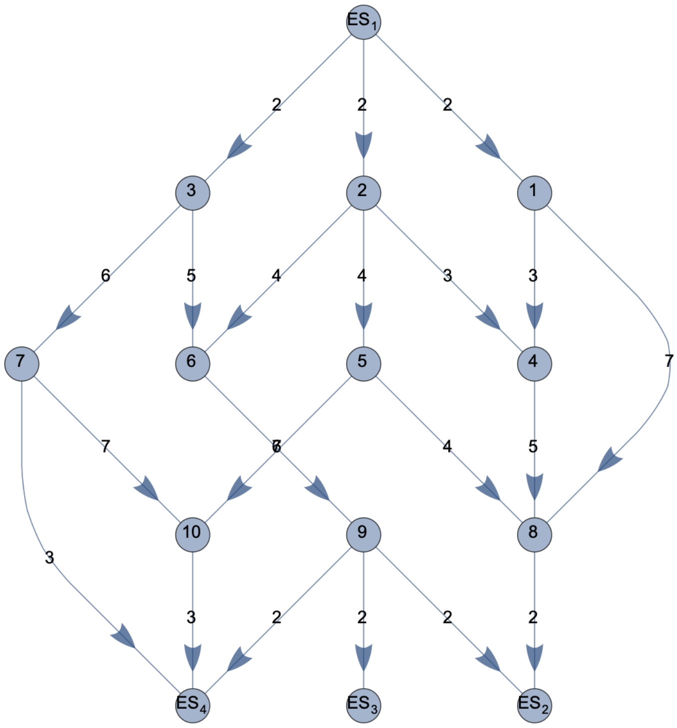
a) Execute the code provided below and take a look at the graph $\mathcal{G}$. This is now the bus system you will use in this task. How many alternative routes are there between ES_1 and ES_2?
Use Mathematica for the following:
b) Find *all paths* from ES_1 to ES_i (for i=2,3,4). How many alternatives are there from ES_1 to ES_2, ES_3, and ES_4?
c) Find *the shortest path* from ES_1 to ES_i (for i=2,3,4).
d) Find the cost of the shortest paths in c). ["cost" is the sum of weights assigned to each road in

the graph as given in the code (third argument of DirectedEdge[]), e.g., representing the driving time.]

e) Plot the shortest paths in the graph $\mathcal{G}$ with the cost of each road/link (Use “HighlightGraph[..]”). If the busses uses these routes only, how does these routes cover the bus stops?

```
In[•]:= (* Build data structure of graph -
 directed edges (going eastbound with weights *)
v = {ES# & /@ Range[4], S# & /@ Range[10]} // Flatten;
(* "hand-coded" set of directed edges with weights:
  DirectedEdge[from,to,weight] *)
eset = {{ES1, 1, 2}, {ES1, 2, 2}, {ES1, 3, 2}, {1, 4, 3},
   {1, 8, 7}, {2, 4, 3}, {2, 5, 4}, {2, 6, 4}, {3, 6, 5}, {3, 7, 6},
   {4, 8, 5}, {5, 8, 4}, {5, 10, 6}, {6, 9, 7}, {7, 10, 7}, {7, ES4, 3},
   {8, ES2, 2}, {9, ES2, 2}, {9, ES3, 2}, {9, ES4, 2}, {10, ES4, 3}};
(* constructe the set of (directed) edges with weights *)
e = DirectedEdge[Part[#, 1], Part[#, 2], Part[#, 3]] & /@ eset;
(* constructe the set of (directed) edges without weights *)
k = e[[#, 1]] ⟷ e[[#, 2]] & /@ Range[1, Length[e]];
(* constructe the set of weights *)
w = e[[#, 3]] & /@ Range[1, Length[e]];
(* constructe the set of edge labels *)
ℓ = ( k[[#]] → w[[#]]) & /@ Range[1, Length[e]];

(* Plot graph *)
𝒢 = Graph[k,
   VertexLabels → Placed["Name", Center],
   VertexSize → Medium,
   EdgeShapeFunction → "FilledArcArrow",
   EdgeLabels → ℓ,
   EdgeWeight → w];
Print["a) Graph, 𝒢=", Show[𝒢, ImageSize → Medium]];
```

a) Graph, $\mathcal{G}$=



a) Using Mathematica functions, we discover that there are six alternative routes in total between ES_1 and ES_2.

In[ ]:=

```
pathsToES2 = FindPath[𝒢, ES1, ES2, ∞, All];
numPathsToES2 = Length[pathsToES2];
Print["ES1 -> ES2:"];
Do[Print["Path ", i, ": ", pathsToES2[[i]]], {i, Length[pathsToES2]}];
```

```
ES1 -> ES2:
Path 1: {ES1, 1, 8, ES2}
Path 2: {ES1, 3, 6, 9, ES2}
Path 3: {ES1, 2, 6, 9, ES2}
Path 4: {ES1, 2, 5, 8, ES2}
Path 5: {ES1, 2, 4, 8, ES2}
Path 6: {ES1, 1, 4, 8, ES2}
```

b) Using Mathematica functions, we discover all paths from ES_1 to ES_i:

In[ ]:=
```
Print["ES₁ -> ES₂:"];
Do[Print["Path ", i, ": ", pathsToES₂[[i]]], {i, Length[pathsToES₂]}];
Print["\n"];

pathsToES₃ = FindPath[𝒢, ES₁, ES₃, ∞, All];
numPathsToES₃ = Length[pathsToES₃];
Print["ES₁ -> ES₃:"];
Do[Print["Path ", i, ": ", pathsToES₃[[i]]], {i, Length[pathsToES₃]}];
Print["\n"];

pathsToES₄ = FindPath[𝒢, ES₁, ES₄, ∞, All];
numPathsToES₄ = Length[pathsToES₂];
Print["ES₁ -> ES₄:"];
Do[Print["Path ", i, ": ", pathsToES₄[[i]]], {i, Length[pathsToES₄]}];
```

```
ES₁ -> ES₂:
Path 1: {ES₁, 1, 8, ES₂}
Path 2: {ES₁, 3, 6, 9, ES₂}
Path 3: {ES₁, 2, 6, 9, ES₂}
Path 4: {ES₁, 2, 5, 8, ES₂}
Path 5: {ES₁, 2, 4, 8, ES₂}
Path 6: {ES₁, 1, 4, 8, ES₂}

ES₁ -> ES₃:
Path 1: {ES₁, 3, 6, 9, ES₃}
Path 2: {ES₁, 2, 6, 9, ES₃}

ES₁ -> ES₄:
Path 1: {ES₁, 3, 7, ES₄}
Path 2: {ES₁, 3, 7, 10, ES₄}
Path 3: {ES₁, 3, 6, 9, ES₄}
Path 4: {ES₁, 2, 6, 9, ES₄}
Path 5: {ES₁, 2, 5, 10, ES₄}
```

In total we have 13 different paths between ES_1 to ES_i .

c) Using Mathematica functions, we find the shortest path from ES_1 to ES_i :

```
shortestPaths = {};
Do[
  shortestPath = FindShortestPath[𝒢, ES1, ESi];
  Print["Shortest path ES1 -> ", ESi, ": ", shortestPath] ×
   AppendTo[shortestPaths, shortestPath];
  , {i, 2, 4}
 ];
```

```
Shortest path ES1 -> ES2: {ES1, 1, 8, ES2}
Shortest path ES1 -> ES3: {ES1, 2, 6, 9, ES3}
Shortest path ES1 -> ES4: {ES1, 3, 7, ES4}
```

d) Using Mathematica functions, we find the cost of the shortest path from ES_1 to ES_i:

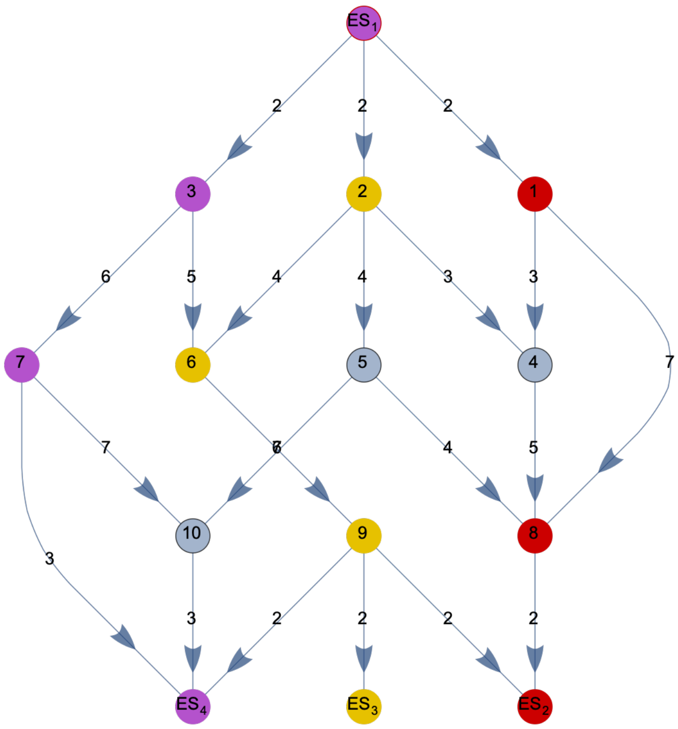
In[∘]:=
```
Do[
  Print["Shortest path cost ES1 -> ", ESi, ": ", GraphDistance[𝒢, ES1, ESi]];
  , {i, 2, 4}
 ];
```

```
Shortest path cost ES1 -> ES2: 11.
Shortest path cost ES1 -> ES3: 15.
Shortest path cost ES1 -> ES4: 11.
```

e) Using Mathematica function, the shortest paths in the graph is highlighted. We can see that the combination of the shortest paths does not cover all of the stops in the system. This means that if the buses were to only use those routes, there are some stops and passengers that never would get served - which probably isn't the best way to go.

In[∘]:=
```
HighlightGraph[𝒢, shortestPaths]
```

Out[∘]=



Useful functions for IV.C.2

In[∘]:=
```
(* Summarise the average number in the system along the "path" - assuming an M/M/1 que
𝒩[path_:List]:=Module[{},
Plus@@Flatten[{QueueProperties[QueueingProcess[λ#,M],"MeanSystemSize"] &/@ path}]
]

(* Returns the route with the highest number of waiting passengers *)
ℳ[path_]:=Module[{Ntmp,wh,pos},
(* Determine the total number of passengers along each of the routes *)
Ntmp= 𝒩[#]&/@ path /. Λ;
(* Select the route with the highest number of passengers *)
wh=Select[Ntmp,#≥Max[Ntmp]&];
pos=Position[Ntmp,wh⟦1⟧]⟦1⟧;
(* Return the selected path and the corresponding number of passengers *)
{path⟦pos⟦1⟧⟧, 𝒩[path⟦pos⟦1⟧⟧]/. Λ}
]

(* Returns a random route with the number of waiting passengers *)
ℳR[path_]:=Module[{wh,pos},
(* Select the route randomly among the alternatives in path *)
pos=RandomInteger[{1,Length[path]}];
(* Return the selected path and the corresponding number of passengers *)
{path⟦pos⟧, 𝒩[path⟦pos⟧]/. Λ}
]
```

IV.C.2*

After a DDOS attack on the on the bus company's server you don't know the state (number/waiting time of passengers) at each station. The busses can still run, but you have to select the next route randomly. In this task, you are going to determine the consequence of that you must select the route randomly and not optimally.

First, we need a metric to quantify how good a route is. Let's assume that this is the sum of passengers at each station along the route (which can be served by this bus). Each bus stop is an M/M/1 queue with the arrival rate defined in the set Λ below. The service rate at each bus stop is *M=3*. Above you find three functions ($\mathcal{N}$[ ], $\mathcal{M}$[ ], $\mathcal{M}$R[ ]) that can be helpful in obtaining the number of passengers waiting at each bus stop, and to summarize the total number along a predefined (the one with most passengers) and along a random route.

a) Find the routes between $ES_1$to ES_i(i=2,3,4) with the highest total number of passengers waiting. [you may use the function $\mathcal{M}$[] from above]
b) Now, assume that the bus coordination system is attacked and out of service. Sample 1000 bus trips. [you may use the function $\mathcal{M}$R[] from above].
c) Obtain the relative difference in number of passengers handled by informed and by random selection of routes, for each of the three pair of end stations: $ES_1$to ES_i(i=2,3,4). Plot the relative difference, with the average and uncertainty of the estimator (choose your *uncertainty metric*, check the textbook for alternatives).

In[ ]:=
```
(* Parameters*)
M = 0.3;
Λ = {λ1 → 0.114, λ2 → 0.176, λ3 → 0.154, λ4 → 0.120, λ5 → 0.181,
    λ6 → 0.175, λ7 → 0.238, λ8 → 0.247, λ9 → 0.123, λ10 → 0.123};
```

a) The routes between $ES_1$to ES_i with the highest total number of passengers waiting are printed below.

In[ ]:=
```
stopsToES2 = {};
stopsToES3 = {};
stopsToES4 = {};

Do[
    Do[
        path = pathsToESi[[j]];
        path = Drop[path, 1];
        path = Drop[path, -1];
        AppendTo[stopsToESi, path],
    {j, 1, Length[pathsToESi]}
    ],
{i, 2, 4}
];

Do[
    Print["Route from ES1 -> " ESi, " with the highest number of passengers: ", M[stops
    {i, 2, 4}
];
```

```
Route from ES1 -> ES2 with the highest number of passengers: {{2, 5, 8}, 7.60074}
Route from ES1 -> ES3 with the highest number of passengers: {{2, 6, 9}, 3.51427}
Route from ES1 -> ES4 with the highest number of passengers: {{3, 7, 10}, 5.58842}
```

b) Sampling is done in cell below. Chose to only print the first five samples for practical reasons.

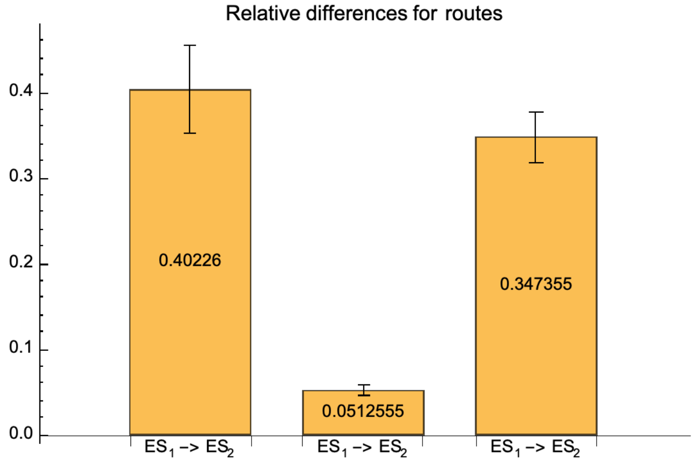
In[ ]:=
```
resultES2 = {};
resultES3 = {};
resultES4 = {};

Do[
  result = Reap[
     Do[
      route = MR[stopsToESj];
      Sow[route],
      {i, 1, 1000}
     ]
    ][[2, 1]];
  Switch[j, 2, resultES2 = result, 3, resultES3 = result, 4, resultES4 = result],
  {j, 2, 4}
 ];

Print["First five samples ES1 -> ES2: \n", resultES2[[1 ;; 5]]]
```

```
First five samples ES1 -> ES2:
{{{3, 6, 9}, 3.14971}, {{2, 5, 8}, 7.60074},
  {{3, 6, 9}, 3.14971}, {{2, 6, 9}, 3.51427}, {{2, 5, 8}, 7.60074}}
```

c) The relative difference is visualised in the bar plot below. I chose to use standard error to show the uncertainty in our measurements. The standard error gives an idea of how much the average relative difference might vary due to sampling. This helps us understand how reliable our estimate is by showing both the average difference and the likely range around it, giving more confidence in the results shown in the plot.

```
In[ ]:= informed = {7.60074, 3.51427, 5.58842};

average = {};
Do[
  avg = Mean[resultESi⟦All, 2⟧];
 AppendTo[average, avg],
 {i , 2, 4}
]

relativeDiffs = {};
Do[
  diff = (informed⟦i⟧ - average⟦i⟧) / average⟦i⟧;
 AppendTo[relativeDiffs, diff],
 {i , 1, 3}
]

stdErrors = {};
Do[
  std = StandardDeviation[resultESi⟦All, 2⟧];
 stdErr = std / Sqrt[Length[resultESi]];
 AppendTo[stdErrors, stdErr],
 {i , 2, 4}
]

labels = {"ES1 -> ES2", "ES1 -> ES3", "ES1 -> ES4"};

plotMaterial = {};
Do[
 numbESi = {Around[relativeDiffs⟦i⟧, stdErrors⟦i⟧]};
 AppendTo[plotMaterial, numbESi],
 {i, Length[relativeDiffs]}
]

BarChart[
 plotMaterial,
 ChartLabels → labels,
 PlotLabel → "Relative differences for routes",
 LabelingFunction → (Placed[#, Center] &)
]
```

Out[ ]=



The labels in the plot are wrong as something got locked up, but the correct labeling would be "ES_1 -> ES_2", "ES_1 -> ES_3", "ES_1 -> ES_4".

d) The bar plot shows that the relative difference in passengers handled is much higher for the first and third routes (ES_1 -> ES_2 and ES_1 -> ES_4) than for the second (ES_1 -> ES_3). This indicates that the bus coordination system significantly improves passenger handling on certain routes, especially those with higher or more variable demand. Overall, the coordination system is essential for efficiently managing waiting passengers, particularly on busier routes.

Task IV.D - Reflections

a) Have you used any AI tool? How and for what have you used it, and how useful did you find it?
b) What have you learned from this assignment? Challenges you have been facing, usability of the tool, relevance for you own interest and studies, etc.

a) I have used ChatGPT to help me with some clever Mathematica hacks. Even though standard googling and searching Mathematica docs turned out to be more efficient most of the time, ChatGPT were able to help me a handful of times.

b) I have learned a lot of Mathematica from this lab. I don’t feel like I have been facing a lot of challenges during this lab to be honest. The biggest problem was understanding some of the tasks where I found the phrasing somewhat unclear. After this lab I am a much bigger fan of Mathematica as a tool, and I have learned that it is a very powerful (and cool) tool. As I have said in the previous labs, really like this assignment and have found it surprisingly educational.