Calculation of the mean time to first failure (MTFF) by a modified state diagram

Bjarne E. Helvik; 24 October 2008, Revised 4 november 2011

Introduction

Objective

In Section 7.2.4 in the textbook [1], a method for determining the dependability measures of a system from a continuous time discrete state Markov model (a state diagram) of the system is presented. One of these dependability measures is the mean time to first failure (MTFF). To avoid introducing a new method, which requires heavier mathematics than what most students are able to cope with, we present a method for modifying the state diagram in a way that allows application of the previously introduced method to find the mean up time (MUT). This method is described in details in the textbook [1].

As a step in applying this method a transition from the failure state to the initial state is introduced and which is assigned an arbitrary repair rate q_0. See the figure below. This results in the following expression

$$\mathrm{MTFF} = \frac{1 - p_F^*}{p_F^* \cdot q_0} \tag{1}$$

where p_F^* is an (artificial) unavailability determined from the modified state diagram.

Every time this is lectured, the following question arises: "How can we introduce an arbitrary intensity q_0 in the diagram/model and get a sensible result? The result (1) is heavily dependent on the choice of q_0!? *The objective of this addition to the description in the textbook is to demonstrate that the choice of q_0 has no influence on the result and why this is the case*.

Tool

To perform the necessary analysis, a Mathematica package developed for dependability analysis by state diagrams is applied

In[1]:=

```
<< "/Users/bjarne/Undervisning/ttm4120/tools/mma_state-diagrams/
   StateDiagrams.m"
```

The function used is described in the last section of the memo. The package StateDiagrams.m with a demo notebook is available at http://www.item.ntnu.no/fag/ttm4120/current/mathematica.php

The modified state diagram

Let us start with the state diagram (Markov model) in Figure 1 from the textbook.

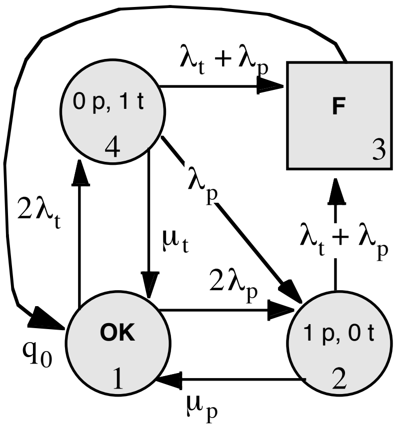


Figure 1

We establish a transition matrix for this model. Note that the matrices used in this tool are the transposed versions of those used in the textbook, see the explanation in the last section of the memo. The transition matrix is defined as a series of sublists, where each list defines the transitions into the corresponding state. The diagonal elements are generated automatically.

In[2]:=

```
(Q16 = SetDiagonal[{{, μp, q0, μt}, {2 λp, , 0, λp}, {0, λp + λt, , λp + λt},
    {2 λt, 0, 0,}}]) // MatrixForm
```

Out[2]//MatrixForm=

$$\begin{pmatrix} -2\lambda_p - 2\lambda_t & \mu_p & q_0 & \mu_t \\ 2\lambda_p & -\lambda_p - \lambda_t - \mu_p & 0 & \lambda_p \\ 0 & \lambda_p + \lambda_t & -q_0 & \lambda_p + \lambda_t \\ 2\lambda_t & 0 & 0 & -2\lambda_p - \lambda_t - \mu_t \end{pmatrix}$$

In addition, we have to identify which states are the working states (circled states), and which are failure states (square states).

In[3]:=

```
working = {True, True, False, True};
```

From the above, we may determine the asymptotic state probabilities and consequently the unavailability.

In[4]:=

```
pF = UnAvailability[ProbStationary[Q16], working] // Simplify
```

Out[4]=

A very large output was generated. Here is a sample of it:

$$\frac{2\left(\lambda_p+\lambda_t\right)\left(2\lambda_p^2+\lambda_t\left(\lambda_t+\mu_p\right)+\lambda_p\left(3\lambda_t+\mu_t\right)\right)}{2\left(\lambda_p+\lambda_t\right)\left(2\lambda_p^2+\lambda_t\left(\lambda_t+\mu_p\right)+\lambda_p\left(3\lambda_t+\mu_t\right)\right)+q_0\left(6\lambda_p^2+\left(\lambda_t+\mu_p\right)\left(3\lambda_t+\mu_t\right)+\lambda_p\left(9\lambda_t+2\mu_p+3\mu_t\right)\right)}$$

Show Less | Show More | Show Full Output | Set Size Limit...

It can be seen that this expression contains the arbitrary repair rate q_0. If the expression for p_F is substituted into the equation for the MTFF in the textbook and by simplifying we find:

In[5]:=

```
MTFF = (1 - pF) / (pF q0) // Simplify
```

Out[5]=

$$\frac{6\lambda_p^2+\left(\lambda_t+\mu_p\right)\left(3\lambda_t+\mu_t\right)+\lambda_p\left(9\lambda_t+2\mu_p+3\mu_t\right)}{2\left(\lambda_p+\lambda_t\right)\left(2\lambda_p^2+\lambda_t\left(\lambda_t+\mu_p\right)+\lambda_p\left(3\lambda_t+\mu_t\right)\right)}$$

The repair rate q_0 is eliminated as we simplify the expression, and the result is independent of the choice of q_0. This will always be the case when the method presented in the textbook is used.

Alternative methods to calculate the MTFF

For the sake of comparison, let us see how the MTFF may be obtained by two other methods. These are based on the original state diagram shown below.

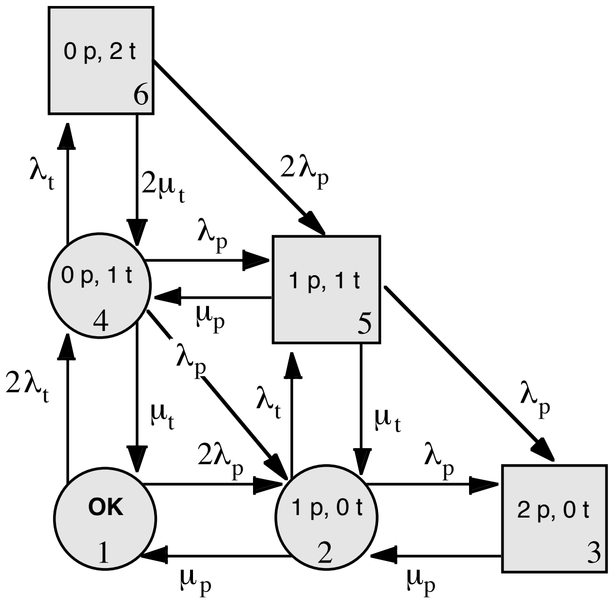


Figure 2

Similar to above we may establish the transition matrix

In[6]:=
```
(Q₂ = SetDiagonal[{{, μp, 0, μt, 0, 0}, {2 λp, , μp, λp, μt, 0},
    {0, λp, 0, 0, λp, 0}, {2 λt, 0, 0, , μp, 2 μt}, {0, λt, 0, λp, , 2 λp},
    {0, 0, 0, λt, 0,}}]) // MatrixForm
```

Out[6]//MatrixForm=

$$\begin{pmatrix} -2\lambda_p - 2\lambda_t & \mu_p & 0 & \mu_t & 0 & 0 \\ 2\lambda_p & -\lambda_p - \lambda_t - \mu_p & \mu_p & \lambda_p & \mu_t & 0 \\ 0 & \lambda_p & -\mu_p & 0 & \lambda_p & 0 \\ 2\lambda_t & 0 & 0 & -2\lambda_p - \lambda_t - \mu_t & \mu_p & 2\mu_t \\ 0 & \lambda_t & 0 & \lambda_p & -\lambda_p - \mu_p - \mu_t & 2\lambda_p \\ 0 & 0 & 0 & \lambda_t & 0 & -2\lambda_p - 2\mu_t \end{pmatrix}$$

and define which states are working and failed states.

In[7]:=
```
working2 = {True, True, False, True, False, False};
```

Calculation by a matrix method

The package StateDiagrams.m implements the method for the calculation of MTFF, etc. described in [2].

Before starting, we nullify the result obtained above (required by *Mathematica*)

In[8]:=
```
MTFF =.
```

and determine the MTFF with the corresponding function in StateDiagrams.m.

In[9]:=
```
MTFF[Q₂, working2] // Simplify
```

Out[9]=

$$\frac{6\lambda_p^2 + (\lambda_t + \mu_p)(3\lambda_t + \mu_t) + \lambda_p(9\lambda_t + 2\mu_p + 3\mu_t)}{2(\lambda_p + \lambda_t)\left(2\lambda_p^2 + \lambda_t(\lambda_t + \mu_p) + \lambda_p(3\lambda_t + \mu_t)\right)}$$

Computation by finding the reliability function

As presented in the textbook [1], the MTFF may be found by integrating the reliability function of the system. This is a computationally far more difficult approach. First we have to determine the reliability function of the system by solving the differential equations to find the transient probabilities (with the failure states made absorbing)

In[31]:=
```
R[τ_] = RelFunc[Q₂, working2, τ]
```

and integrate to find the expectation.

```
MTFF = ∫_0^∞ R[τ] ⅆτ
```

$$MTFF = \int_0^{\infty} R[\tau]\, d\tau$$

However, finding a symbolic solution for $R[\tau]$ is extremely demanding, maybe infeasible, even for a small model like this, and is omitted. Numerical calculation with this method works better, but we may soon run into numerical problems if the system gets larger and the λ/μ ratio is small. This approach to find MTFF is not advised.

Description of Functions

Descriptions of the functions of StateDiagrams.m used in the calculations are given below. The descriptions are included in the package and are obtained by prefixing the function name with a question mark.

```
? SetDiagonal
```

```
SetDiagonal[m] produces a complete transition matrix by introducing
  diagonal elements. m is an incomplete transition matrix
  where m[[j,i]] i ≠ j is the transition rates from  state
  i to j.  NB! i is the matrix column no., j is the row no.
```

```
? ProbStationary
```

```
ProbStationary[m] determines the stationary state probability vector
  for the system defined by the  complete transition matrix m.
```

```
? UnAvailability
```

```
UnAvailability[ps,working] computes the unavailability  for a system
  with state probability vector ps and a  corresponding Boolean
  vector working,  where working[[i]]=True if i is a working state.
```

```
? MTFF
```

```
MTFF[m, WorkingQ, Ini] determines the  Mean Time to First Failure.
  The system is defined by the  complete transition matrix
  m and the working states is  given by the corresponding
  Boolean vector WorkingQ, where WorkingQ[[i]]=True if i
  is a working state.  The system is in state Ini at time
  t=0.  The parameter Ini is optional with default value 1.
```

```
? RelFunc
```

```
RelFunc[m, WorkingQ, t, Ini] determines the reliability
  function  (the probability that the system has not failed
  in [0,t]) at time t. The system is defined by the  complete
  transition matrix m and the working states are  given by the
  corresponding  Boolean vector WorkingQ, where WorkingQ[[i]]=True
  if i is a working state. The system is in state Ini at time
  t=0.  The parameter Ini is optional with default value 1.
```

Generation of the state diagram

The StateDiagrams.m does also contain functionality for generating system models from models of subsytems and to manipulate and plot system models. This section gives an example where the above submodel is obtained and plotted.

 The state model of the DNS server system consist of the composition of to individual and identical servers, having a common repairman/facility as their only dependency. Hence, it may be generated from the model of a singel server with the modification that when there are two servers having permanent failures, the reepair intnesity is still μ_p for the entire system.

Single server model

The model of a single server is

```
(Ω = SetDiagonal[Transpose[{{0, λt, λp}, {μt, 0, λp}, {μp, 0, 0}}]]) //
 MatrixForm
```

Out[11]//MatrixForm=

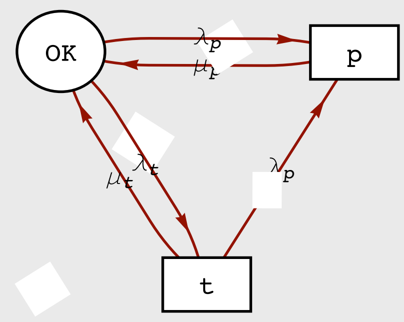
$$\begin{pmatrix} -\lambda_p - \lambda_t & \mu_t & \mu_p \\ \lambda_t & -\lambda_p - \mu_t & 0 \\ \lambda_p & \lambda_p & -\mu_p \end{pmatrix}$$

In[12]:=

```
ElemntLabels = {"OK", "t", "p"};
ElemntWorkQ = {True, False, False};
```

In[14]:=
```
PlotDiagram[Ω, ElemntWorkQ, ElemntLabels]
```

Out[14]=



Dual and multiple server models

We may now generatte the duplicated system from the lecture notes by using the function GenHomogeneousSystem., [EHH07], where the generated model is modified for the single repairman.

In[15]:=
```
? GenHomogeneousSystem
```

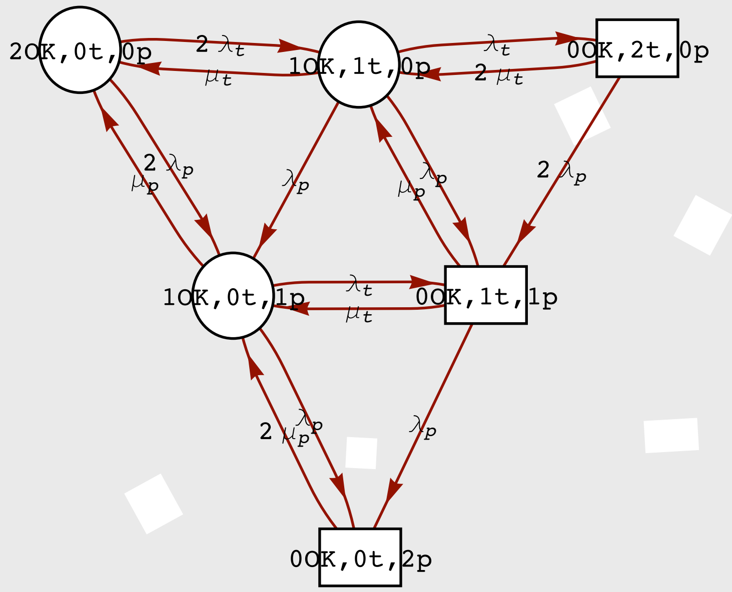
GenHomogeneousSystem[m,
WorkingQ, Labels, n, k, Index] Generates a system
consiting of n transition matrix dependability model represented by a
homogeneous subsystems (elements)
where the parameters are the transition
matrix m, the corresponding Boolean vector WorkingQ, where
WorkingQ[[i]]=True if i is a working state, and the list of state labels
denoted Labels. The system is
working if k–out–of–n subsytems are working.
The function returns {sm, sWorkingQ, sLabels}.
Index is an optional argument,
where the indexing of states are returned,
i.e.,Index[{a,b,..,c}]give the mapping between the number of
subsytems in their local states and the system state no.

```
pyseSyst = GenHomogeneousSystem[Ω, ElemntWorkQ, ElemntLabels, 2, 1];
```

In[18]:=

```
Apply[PlotDiagram, pyseSyst]
```

Out[18]=



It is seen that the gerrated model has an occurence of nμ_p wher in this case $n = 2$ representing more than one repairman. This is modified in the generated model by replacing this traitions with the repair rate for a single repairman, i.e.

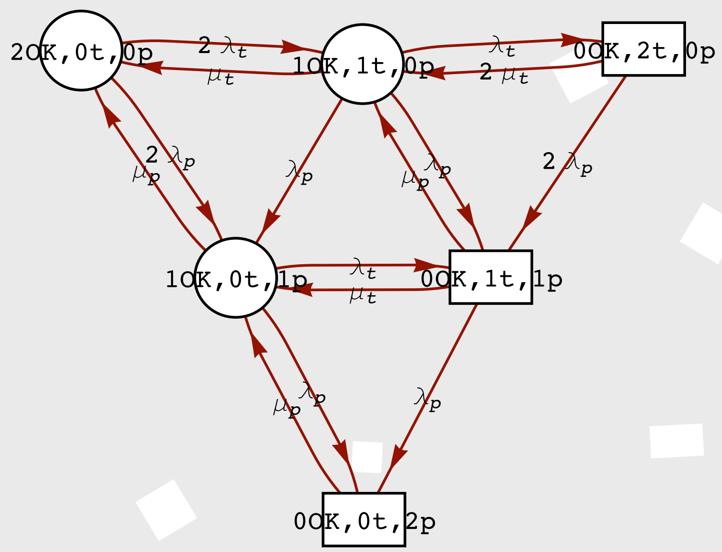
In[21]:=

```
pyseSyst[[1]] = SetDiagonal[pyseSyst[[1]] /. _μp → μp];
```

which yield a model identical to the one in the lecture notes shown above.

In[22]:=

```
Apply[PlotDiagram, pyseSyst]
```

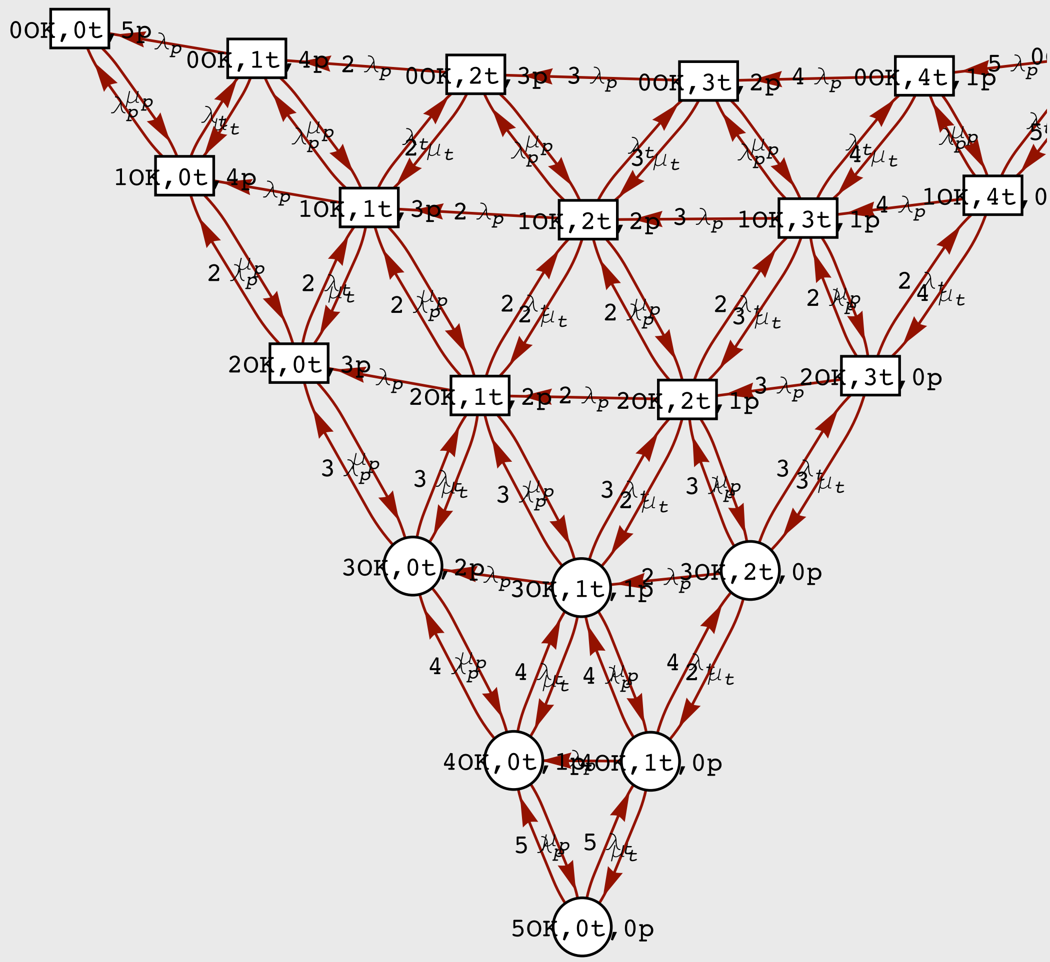


A system with more subsystems, just for the demo. Note that we require that 3-out-of-5 servers should work for the system to be working

In[27]:=

```
pyseSystM = GenHomogeneousSystem[Ω, ElemntWorkQ, ElemntLabels,
   5, 3, cyy];
(pyseSystM[[1]] = SetDiagonal[pyseSystM[[1]] /. _μp → μp]) ;
Apply[PlotDiagram, pyseSystM]
```

Out[29]=



and in this case the MTFF becomes

In[30]:=

```
MTFF[pyseSystM[[1]], pyseSystM[[2]]] // Simplify
```

$$\Big(4700\,\lambda_p^5+5\,\lambda_p^4\,(3901\,\lambda_t+207\,\mu_p+799\,\mu_t)+(3\,\lambda_t+\mu_p)\,(47\,\lambda_t^2+13\,\lambda_t\,\mu_t+2\,\mu_t^2)\,(12\,\lambda_t^2+7\,\lambda_t\,\mu_p+\mu_p\,(\mu_p+\mu_t))+\lambda_p^3\,(32\,101\,\lambda_t^2+140\,\mu_p^2+1009\,\mu_p\,\mu_t+1081\,\mu_t^2+\lambda_t\,(5335\,\mu_p+11\,114\,\mu_t))+\lambda_p^2\,(26\,179\,\lambda_t^3+5\,\mu_p^3+141\,\mu_p^2\,\mu_t+278\,\mu_p\,\mu_t^2+94\,\mu_t^3+\lambda_t^2\,(9002\,\mu_p+10\,845\,\mu_t)+\lambda_t\,(750\,\mu_p^2+2731\,\mu_p\,\mu_t+1770\,\mu_t^2))+\lambda_p\,(10\,575\,\lambda_t^4+\lambda_t^3\,(6253\,\mu_p+4194\,\mu_t)+\mu_p\,\mu_t\,(7\,\mu_p^2+39\,\mu_p\,\mu_t+16\,\mu_t^2)+3\,\lambda_t^2\,(364\,\mu_p^2+754\,\mu_p\,\mu_t+267\,\mu_t^2)+\lambda_t\,(40\,\mu_p^3+342\,\mu_p^2\,\mu_t+363\,\mu_p\,\mu_t^2+54\,\mu_t^3))\Big)\Big/\Big(60\,(\lambda_p+\lambda_t)\,(100\,\lambda_p^5+5\,\lambda_p^4\,(83\,\lambda_t+\mu_p+17\,\mu_t)+\lambda_t^2\,(3\,\lambda_t+\mu_p)\,(12\,\lambda_t^2+7\,\lambda_t\,\mu_p+\mu_p\,(\mu_p+\mu_t))+\lambda_p^3\,(683\,\lambda_t^2+\mu_t\,(7\,\mu_p+23\,\mu_t)+\lambda_t\,(65\,\mu_p+217\,\mu_t))+\lambda_p\,\lambda_t\,(225\,\lambda_t^3+2\,\mu_p\,\mu_t\,(\mu_p+\mu_t)+17\,\lambda_t^2\,(7\,\mu_p+3\,\mu_t)+2\,\lambda_t\,(8\,\mu_p^2+11\,\mu_p\,\mu_t+3\,\mu_t^2))+\lambda_p^2\,(557\,\lambda_t^3+2\,\mu_t^2\,(\mu_p+\mu_t)+\lambda_t^2\,(146\,\mu_p+183\,\mu_t)+\lambda_t\,(5\,\mu_p^2+28\,\mu_p\,\mu_t+27\,\mu_t^2)))\Big)$$

References

[1] Peder Emstad, Poul E. Heegaard, Bjarne E. Helvik and Laurent Paquerau: Dependability and Performance in Information and Communication Systems; Fundamentals, (278 p.), Tapir academic publisher, Aug. 2008.

[2] John A.Buzacott, “Markov Approach to Finding Failure Times of Repairable Systems”, IEEE Transactions on Reliability, Vol.R - 19, No.4, pp. 128 - 133, November 1970.