Exam - Tuesday 30. may 2006

TTM4120 Pålitelige systemer
Dependable systems

Proposed solution

Version 0.3; 19 June 2006; BEH

a) **Availability**

The availability of a system is its ability to provide a set of services at a given instant of time or at any instant within a given time interval.

Instantaneous availability

The instantaneous availability, denoted $A(t)$, which also is called the pointwise availability; is the probability that the system is working at a given instant of time t. Let $I(t)$ take the value "Up" when the system is working, otherwise "Down". Then: $A(t) = P(I(t) = \text{Up})$

For more measures see the textbook section 1.4.2

Reliability

The reliability of a system is its ability to provide uninterrupted service.

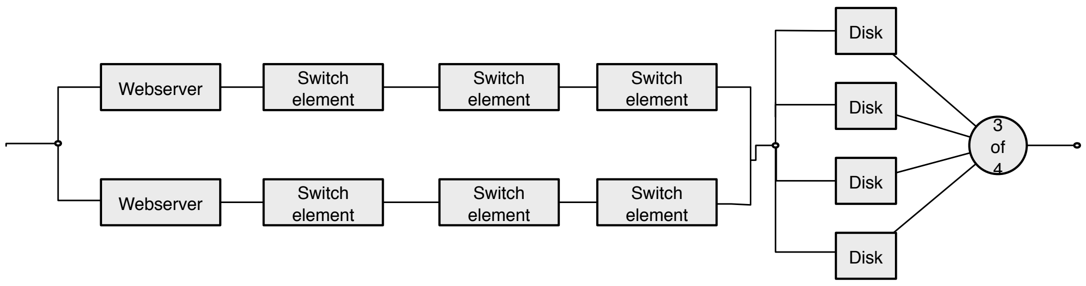
The reliability function of a service provided ba a system $R(t)$ is defined as

$$R(t) = P(T_F > t) \ , \tag{0.1}$$

where T_F is the time from the service starts and until it fails. For more info. see Section 1.4.1, subsection "System in steady state in the textbook.

(The reliability function is *the probability that a system can provide its required services (under stated conditions) for a given time interval)*

b) The availability of the service provided by the system may under the assumptions stated can be derived from the reliability block scheme below.

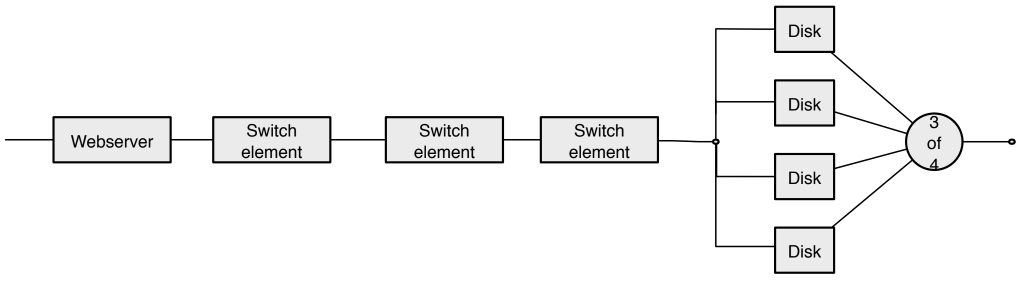


From this we obtain the unavailability U

$$\begin{aligned} U_A &= 1-(1-U_t)(1-U_s)^3 \\ U_{AA} &= U_A^2 \\ A_D &= \sum_{j=3}^{4}\binom{4}{j}(1-U_d)^j(U_d)^{4-j} \\ U &= 1-(1-U_{AA})A_D \end{aligned} \tag{0.2}$$

[In the first draft, the following sub-question was included: Vis hvordan vi kan bestemme midlere nedetid for tjenestene. (Merk detaljerte uttrykk kreves ikke, kun at det vises hvordan man går frem.) Removed to reduce the exam workload.]

c) An important issue in answering this question is that the service provided to a user must be delivered on the web server on which the user is logged on. Hence, the corresponding reliability block diagram becomes



However, the various system components are repaired (given from the formulation of the exam) and the scheme method cannot be used. (e.g. it is impossible at an arbitrary point in time to reflect the disk system is working and has not previously failed)

In this case, only the disk subsystem has redundancy and which repair will influence the reliability of the service provided. Hence, if it is assumed that this subsystem does not fail, any failure of the four remaining system element will will cause the service to fail, and $R(t) = \exp(-(\lambda_t + 3\lambda_s)t)$

[Is this an upper or lower bound on the reliability function provided?]

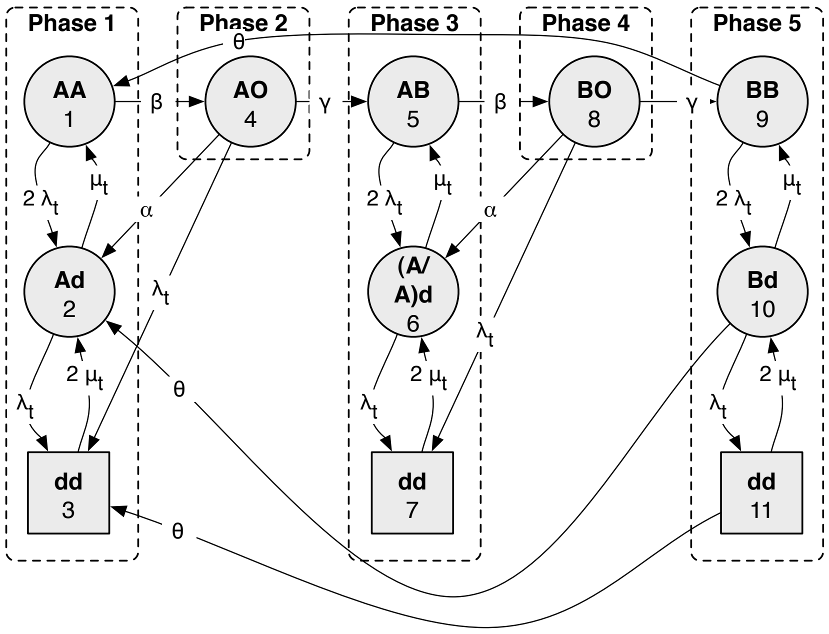
d) The main issue here is to recognize that the state associated with the provisioning of service to a specific user is available on only one server. When this server fails, the service fails.

Hence, the state of the user must be maintained during server failures. Since, (from the text) replication of each server or running the servers in HW synchronism is ruled out, two (or more?) options remain:

- To introduce middleware (e.g. Jgroup, FT Corba) which maintain synchronous copies of the state space on both servers. This allows the other server to take over the work if one failes. This option is close to the mandatory lab. assignment. Variants are:
  - to use passive replication (stand-by) where individual processes synchronizes, or
  - active replication, which may be an "overkill" for the actual application.
- To write the user state to file/database (stable storage) and fetch this for every interaction with the user (i.e. using a stateless server.). This scheme provides loadsharing on a per interaction basis. (This option is not discussed in the written curriculum, but explained/mentioned during lectures.)
- … more …?.

[At the exam in 2006, most of the students assumed that the problem was that both servers could not reach both interconnect switches, and hence, introduced sectioned redundancy to rectify the problem. This will improve the reliability if the switches have a failure rate substantially higher than the web server, but does not rectify the basic problem pointed out above. The two options above will also enable us to tolerate switch faults. (Note also that redesigning the switches in this way requires new switches (2x2) and does not comply with the premises of the question)

e) The state diagram may be as the one below



Here:

A: denotes one (arbitrary) server with the old software version.

B: denotes one (arbitrary) server with the new software version

A/B: denotes a server with either the new or the old software version

O: denotes one (arbitrary) server under upgrade.

d: denotes one (arbitrary) failed server

State numbers are added for indexing and for making the **dd** states unique. Circular states are working states, squared are down states.

f) We see from the diagram that the fault free upgrade sequence of states are 1, 4, 5, 8 and 9. By using the Markov properties we obtain.(keep in mind that state 9 is not a part of the upgrade):

- The probability of an upgrade without failures: $\left(\frac{\beta}{\beta + 2\lambda_t}\right)^2 \left(\frac{\gamma}{\gamma + \alpha + \lambda_t}\right)^2$
- The expected duration of an upgrade without failures: $\left(\frac{2}{\beta + 2\lambda_t}\right) + \left(\frac{2}{\gamma + \alpha + \lambda_t}\right)$

g) From the diagram we see

$$\begin{bmatrix} \lambda_{11} & \lambda_{21} \\ \lambda_{12} & \lambda_{22} \end{bmatrix} = \begin{bmatrix} -2\lambda_t - \beta & \mu_t & 0 & 0 \\ 2\lambda_t & -\lambda_t - \mu_t & 2\mu_t & \alpha \\ 0 & \lambda_t & -2\mu_t & \lambda_t \\ \beta & 0 & 0 & -\lambda_t - \alpha - \gamma \end{bmatrix} . \tag{0.3}$$

h) Define the state vector, where the elements are the probabilities of the states in the diagram. $\underline{P} = \{p_1, \ldots, p_{11}\}^T$.

The normality condition, $\{1, \ldots, 1\} \cdot \underline{P} = 1$,is introduced by replacing one line in the set of linear equations with this. (Since it is only one state in phase 2 (and 4)from, i.,e, one row in the submatrixes, it is easiest to replace these. Note that this “trick” row 2 and 4 is not required; replacement of any row is ok).with is not

$$\Lambda^* = \begin{bmatrix} \lambda_{11} & \lambda_{21} & 0 & 0 & \lambda_{51} \\ 1 & 1 & 1 & 1 & 1 \\ 0 & \lambda_{23} & \lambda_{33} & \lambda_{43} & 0 \\ 0 & 0 & \lambda_{34} & \lambda_{44} & 0 \\ 0 & 0 & 0 & \lambda_{45} & \lambda_{55} \end{bmatrix} \tag{1}$$

Hence, the steady state equations may be obtained from

$$\Lambda^* \underline{P} = \underline{B} \tag{2}$$

where $\underline{B} = \{0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0\}^T$ and since only states 3, 7 and 11 are down states, the unavailability becomes

$$U = p_3 + p_7 + p_{11} \tag{3}$$

i) We see that the requested time is the time from the system starts in state 1 and until it reaches state 9 for the first time. We apply the computation technique for the reliability function and 1) makes state 9 absorbing, and 2) remove the uninteresting states 10 and 11. This yields the transition matrix

$$\Lambda^{\dagger} = \begin{bmatrix} \lambda_{11} & \lambda_{21} & 0 & 0 & 0 \\ \lambda_{12} & \lambda_{22} & 0 & 0 & 0 \\ 0 & \lambda_{23} & \lambda_{33} & \lambda_{43} & 0 \\ 0 & 0 & \lambda_{34} & \lambda_{44} & 0 \\ 0 & 0 & 0 & \gamma & 0 \end{bmatrix} \tag{0.4}$$

The transient state probability vector is defined as. $\underline{P}(t) = \{p_1(t), \ldots, p_9(t)\}^T$. The initial value. $\underline{P}(0) = \{1, 0, 0, \ldots, 0\}^T$ since both processors are up when the upgrade starts.

These probabilities may be found by solving the set of first order linear differential equations with constant coefficients with the given initial condition

$$\frac{d}{dt}\underline{P}(t) = \Lambda^{\dagger}\underline{P}(t) \tag{0.5}$$

Since $T_o > t$ when the system has not yet reached state 9 we have that

$P(T_o > t) = 1 - p_9(t)$