3 Constructing and Solving Markov Processes

3.1 Stochastic Processes

Formally, a stochastic model is one represented as a *stochastic process*; a stochastic process is a set of random variables $\{X(t), t \in T\}$. T is called the *index set* and in all the models we consider T will be taken to represent time. Since we consider continuous time models $T = \mathbb{R}$, the set of real numbers. The *state space* of the process is the set of all possible values that the random variables $X(t)$ can assume. Each of these values is called a *state* of the process. These states will correspond to our intuitive notion of states within the system represented, although there will not necessarily be a one-to-one relation. Any set of instances of $\{X(t), t \in T\}$ can be regarded as a path of a particle moving randomly in the state space, S, its position at time t being $X(t)$. These paths are called *sample paths* or *realisations* of the stochastic process.

For example, if we consider the mobile phone antenna with frequency division multiplexing again, we might associate one random variable with each frequency and associate the values 1 and 0 with the frequency being used or not. At any time the random variables representing the system will be the set of random variables representing each of the frequencies. Thus the state of the system is an n-tuple, where n is the number of frequencies and the entry in each position represents the state of the individual frequency. The state will change whenever a new connection is made or an existing connection is terminated. Alternatively, we might consider the stochastic process characterised by a single random variable, M, which records the number of frequencies in use. Now a single state at the stochastic process level will correspond to several states of the system depending on which frequencies are in use. This example shows how the degree of abstraction influences the modelling process.

The stochastic processes we will be considering during the first half of the course will all have the following properties:

$\{X(t)\}$ is a Markov process. This implies that $\{X(t)\}$ has the *Markov* or *memoryless property*: given the value of $X(t)$ at some time $t \in T$, the future path $X(s)$ for $s > t$ does not depend on knowledge of the past history $X(u)$ for $u < t$, i.e. for $t_1 < \cdots < t_n < t_{n+1}$,

$$\Pr(X(t_{n+1}) = x_{n+1} \mid X(t_n) = x_n, \ldots, X(t_1) = x_1) = \Pr(X(t_{n+1}) = x_{n+1} \mid X(t_n) = x_n)$$

$\{X(t)\}$ is irreducible. This implies that all states in S can be reached from all other states, by following the transitions of the process.

$\{X(t)\}$ is stationary: for any $t_1, \ldots t_n \in T$ and $t_1 + \tau, \ldots, t_n + \tau \in T$ $(n \geq 1)$, then the process's joint distributions are unaffected by the change in the time axis and so,

$$F_{X(t_1+\tau)\ldots X(t_n+\tau)} = F_{X(t_1)\ldots X(t_n)}$$

$\{X(t)\}$ is time homogeneous: the behaviour of the system does not depend on when it is observed. In particular, the transition rates between states are independent of the time at which the transitions occur. Thus, for all t and s, it follows that
$\Pr(X(t + \tau) = x_k \mid X(t) = x_j) = \Pr(X(s + \tau) = x_k \mid X(s) = x_j)$.

3.2 Markov Processes

In this course, our primary objective with respect to a Markovian model will be to calculate the probability distribution of the random variable $X(t)$ over the state space S, as the system settles into a regular pattern of behaviour. This is termed the *steady state probability distribution*. From this probability distribution we will derive performance measures based on subsets of states where some condition holds. Markov processes are widely used because it is well-understood how to solve them.

The memoryless property means that once we have arrived at a particular state the future behaviour is always the same regardless of how we arrived in the state. The solution of Markov processes is closely related to their representation by matrices. Finding the average behaviour of the model then corresponds to solving a simple matrix equation. This will be discussed in detail in the remainder of this note.

Like any stochastic process, a Markov process is characterised by a random variable $X(t)$, indexed over time t, and a state space S such that $X(t) \in S$ for all t. The dynamic behaviour of the system being modelled is represented by the transitions between these states, and times spent in states, *sojourn times*. In general, these holding times represent the periods while processing is occurring in the system, the transitions represent events in the system.

If a state $i \in S$ is entered at time t and the next state transition takes place at time $t + T$, T is the sojourn time in state i. By the Markov property, at any time point τ, the distribution of the time until the next change of state is independent of the time of the previous change of state. In other words, sojourn times are memoryless. Since the only probability distribution function which has this property is the exponential distribution this tells us that the sojourn time in any state of a Markov process is an exponentially distributed random variable. Hence at time τ, the probability that there is a state transition in the interval $(\tau, \tau + dt)$ is $q_i dt + o(dt)$, where q_i is the parameter of the exponential distribution of the sojourn time in state i.

Suppose that when a transition out of state i occurs, the new state is j with probability p_{ij}. By the Markov property, this must only depend on i and j. Thus we have, for $i \neq j$, $i, j \in S$,

$$\Pr(X(t + dt) = j \mid X(t) = i) = q_{ij} dt + o(dt)$$

where the $q_{ij} = q_i p_{ij}$, by the decomposition property. The q_{ij} are called the *instantaneous transition rates*, and if the average time delay before a transition from i to j is exponentially distributed with parameter μ, then $q_{ij} = \mu$.

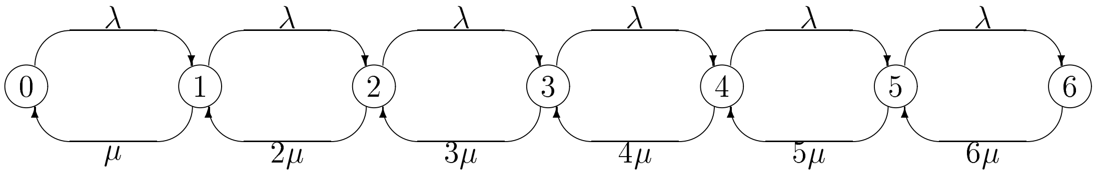
In practice, we are usually given the q_{ij} because we will know the parameter of the distribution governing the time delay associated with the event which is represented by the state transition. For example, we might be told that the average duration of a call using the mobile phone antenna is $1/\mu$ seconds. If we are using Markovian analysis, we then *choose to assume* that the duration is exponentially distributed with parameter μ.

From the q_{ij} we can derive the *exit rate*, q_i, and the transition probabilities, p_{ij}, as follows. The exit rate is the rate at which the system leaves state i, i.e. it is the parameter of the exponential distribution governing the sojourn time. This will be the minimum of the delays until any of the possible transitions occurs. Thus, by the superposition property it is the sum of the individual transition rates, i.e. $q_i = \sum_{j \in S} q_{ij}$. The transition probability p_{ij} is the probability, given that a transition out of state i occurs, that it is the transition to state j. By the definition of conditional probability, this is $p_{ij} = q_{ij}/q_i$.

3.2.1 State transition diagrams

For small Markov processes the simplest way to represent the process is often in terms of its state transition diagram. In such a diagram we represent each state of the process as a node in a graph. The arcs in the graph represent possible transitions between states of the process. The arcs are labelled by the *transition rates* between states. Since every transition is assumed to be governed by an exponential distribution, the rate of the transition will be the parameter of the distribution. Irreducibility implies that the state transition diagram must be strongly connected.

Example: Consider again the mobile phone antenna taking the simplest representation of the states of the system, i.e. the number of frequencies in use. Then, if there are 6 frequencies, the state space is $\{0, 1, 2, 3, 4, 5, 6\}$. The only possible transition from state 0 is to state 1, caused by a new connection. Similarly, from state 6 the only possible transition is to state 5, caused by a frequency being released. For any other state, j, $1 \leq j \leq 5$ there are two possible transitions, to states $j+1$ or $j-1$, caused by a new connection or release of an old one, respectively. Let us assume that the arrival of calls form a Poisson stream with parameter λ: the rate at which $j \to j+1$ transitions occur is λ. As above, we assume that the mean duration of a connection is $1/\mu$, so that the rate at which release transitions occur on each frequency is μ. Then the state transition diagram for this process is:



3.2.2 Generator matrices

A Markov process with n states can also be characterised by its $n \times n$ *infinitesimal generator matrix*, $\mathbf{Q}$. The entry in the jth column of the ith row of the matrix ($j \neq i$) will be q_{ij}, the instantaneous transition rate. In other words, it will be the sum of the parameters labelling arcs connecting nodes i and j in the state transition diagram. The diagonal elements are chosen to ensure that the sum of the elements in every row is zero, i.e. $q_{ii} = -\sum_{j \in S, j \neq i} q_{ij}$.

Example: For the mobile phone antenna example, whose state transition diagram is shown in the previous section, the infinitesimal generator matrix is:

$$\mathbf{Q} = \begin{pmatrix} -\lambda & \lambda & 0 & 0 & 0 & 0 & 0 \\ \mu & -(\mu+\lambda) & \lambda & 0 & 0 & 0 & 0 \\ 0 & 2\mu & -(2\mu+\lambda) & \lambda & 0 & 0 & 0 \\ 0 & 0 & 3\mu & -(3\mu+\lambda) & \lambda & 0 & 0 \\ 0 & 0 & 0 & 4\mu & -(4\mu+\lambda) & \lambda & 0 \\ 0 & 0 & 0 & 0 & 5\mu & -(5\mu+\lambda) & \lambda \\ 0 & 0 & 0 & 0 & 0 & 6\mu & -6\mu \end{pmatrix}$$

3.3 Steady state distribution

Performance analysis is usually concerned with the behaviour of systems over an extended period of time. The running time of the system will be very large relative to the individual inter-event times within the system. From a modelling point of view this is important because we may have to choose an initial state of our model arbitrarily. Considering the long term probability distribution will overcome any bias introduced by the chosen starting state. This will have serious impact on how we use models when we consider simulation models, because each *run* of the model is a particular *sample path* of the model over its state space. For Markovian models we avoid the problems of bias because we do not consider individual sample paths but the probability distribution over the states of the process which we can deduce knowing the transition rates between states.

We assume that when the effects of initial bias have been overcome, the model, and the system, will enter into *equilibrium* or *steady state* behaviour. This does not mean that the model has entered one state and no longer makes transitions. It does mean that the model is assumed to exhibit regularity and predictability in its behaviour over the state space. Thus, the probability distribution of the random variable $X(t)$ over the state space S, will be no longer changing.

Let us denote by $\pi_k(t)$ the probability that if we observe the Markov process at time t it will be in state x_k, for $x_k \in S$. Equilibrium has been reached when, for all states $x_k \in S$, $\pi_k(t+\tau) = \pi_k(t)$, i.e. the time at which we observe the model does not influence the probability that it is in a particular state. Thus we denote the *steady state probabilities* without the time variable, e.g. $\boldsymbol{\pi_k}$, for the probability that the model is in state x_k.

The steady state distribution is useful to us because the following theorem tells us that, for the types of model we are considering, the steady state distribution is exactly the same as the long term probability distribution over states.

Theorem: A steady state probability distribution, $\{\boldsymbol{\pi_k}, x_k \in S\}$, exists for every time homogeneous, finite, irreducible Markov process. Moreover, this distribution is the same as the limiting distribution

$$\boxed{\lim_{t\to\infty} \Pr(X(t) = x_k \mid X(0) = x_0) = \boldsymbol{\pi_k}}$$

3.3.1 Global Balance Equations

In steady state, $\boldsymbol{\pi_i}$ is the proportion of time that the process spends in state x_i. Recall that the transition "rate" q_{ij} is the instantaneous probability that the model makes a transition from state x_i to state x_j. Thus, in an instant of time, the probability that a transition will occur from state x_i to state x_j is the probability that the model was in state x_i, $\boldsymbol{\pi_i}$, multiplied by the transition rate q_{ij}. This is called the *probability flux* from state x_i to state x_j.

When the model is in steady state, in order to maintain the equilibrium, we must assume that the total probability flux out of a state is equal to the total probability flux into the state. If this were not the case the distribution over states would change. So, for any particular state x_i, this implies

$$\boxed{\underbrace{\boldsymbol{\pi_i} \times \sum_{x_j \in S, j \neq i} q_{ij}}_{\text{flux }\textbf{out}\text{ of } x_i} = \underbrace{\sum_{x_j \in S, j \neq i} (\boldsymbol{\pi_j} \times q_{ji})}_{\text{flux }\textbf{into}\ x_i}}$$

The collection of these equations for all states $x_i \in S$ is called the *global balance equations*. If we recall that the diagonal elements of the infinitesimal generator matrix $\mathbf{Q}$ are the negative sum of the other elements in the row (i.e. $q_{ii} = -\sum_{x_j \in S, j \neq i} q_{ij}$) we can rearrange the equation above to be:

$$\sum_{x_j \in S} \boldsymbol{\pi_j} q_{ji} = 0.$$

Expressing the values $\boldsymbol{\pi_i}$ as a row vector $\boldsymbol{\pi}$, we can write this as a matrix equation:

$$\boxed{\boldsymbol{\pi}\ \mathbf{Q} = 0}$$

The $\boldsymbol{\pi_i}$ are unknown; they are the values we wish to find since they are the probability distribution. If there are n states in the state space there are n such equations in n unknowns. Unfortunately, this collection of equations is irreducible: we need another equation in order to solve the equations and find the unknowns. Fortunately, since $\{\boldsymbol{\pi_i}\}$ is a probability distribution we also know that the *normalisation condition* holds:

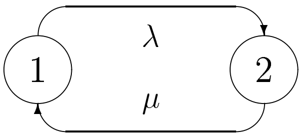
$$\boxed{\sum_{x_i \in S} \boldsymbol{\pi_i} = 1}$$

With these $n + 1$ equations we can solve to find the n unknowns, $\{\boldsymbol{\pi_i}\}$.

Example: Consider a system with multiple CPUs, each with its own private memory, and one common memory which can be accessed only by one processor at a time. The CPUs execute in private memory for a random time before issuing a common memory access request. Assume that this random time is exponentially distributed with parameter λ (the average time a CPU spends executing in private memory between two common memory access requests is $1/\lambda$). The common memory access duration is also assumed to be exponentially distributed, with parameter μ (the average duration of a common memory access is $1/\mu$). If the system has only one processor, it has only two states:

1. The processor is executing in its private memory;
2. The processor is accessing common memory.

The system behaviour can be modelled by a 2 state Markov process whose state transition diagram and generator matrix are as shown below:



$$\mathbf{Q} = \begin{pmatrix} -\lambda & \lambda \\ \mu & -\mu \end{pmatrix}$$

If we consider the probability flux in and out of state 1 we obtain: $\boldsymbol{\pi_1}\ \lambda = \boldsymbol{\pi_2}\mu$
Similarly, for state 2 [1]: $\boldsymbol{\pi_2}\ \mu = \boldsymbol{\pi_1}\lambda$

Finally, we know from the normalisation condition that: $\boldsymbol{\pi_1} + \boldsymbol{\pi_2} = 1$

It follows that the steady state probability distribution is $\boldsymbol{\pi} = \left(\frac{\mu}{\mu+\lambda}, \frac{\lambda}{\mu+\lambda}\right)$. From this information we can deduce, for example that the probability that the processor is executing in its private memory is $\mu/(\mu+\lambda)$.

3.3.2 Solving the global balance equations

Once we are considering models which have more than a few of states it is unlikely that we want to solve the equations ourselves by hand. Instead we take advantage of the matrix equation formulation of the global balance equations and use suitable software packages. Any linear algebra package will solve matrix equations of the form $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A}$ is an $n \times n$ matrix, $\mathbf{x}$ is a column vector of n unknowns, and $\mathbf{b}$ is a column vector of n values. There are two problems which have to be resolved before we can take advantage of such software:

1. Our global balance equation comes in the "wrong" form $\boldsymbol{\pi}\ \mathbf{Q} = 0$: the unknowns, $\boldsymbol{\pi}$ are given as a row vector not a column vector. This problem is resolved by transposing[2] the equation, i.e. $\mathbf{Q}^T\boldsymbol{\pi} = 0$, where the right hand side is now a column vector of zeros, rather than a row vector.
2. The second problem is the redundancy amongst the global balance equations—even after transposing there is insufficient information to solve for the unknowns. As already noted, we need the additional information of the normalisation condition. This problem is resolved by replacing one of the global balance equations by the normalisation condition. In the transposed matrix this corresponds to replacing one row by a row of 1's. We usually choose the last row and denote the modified matrix $\mathbf{Q}_n^T$. Similarly we change the "solution" vector, which was all zeros to be a column vector with 1 in the last row, and zeros everywhere else. We denote this vector, $\mathbf{e}_n$.

Now we can use any linear algebra solution package, such as `maple` or `xmaple` to solve the resulting equation[3]:

$$\mathbf{Q}_n^T\boldsymbol{\pi} = \mathbf{e}_n$$

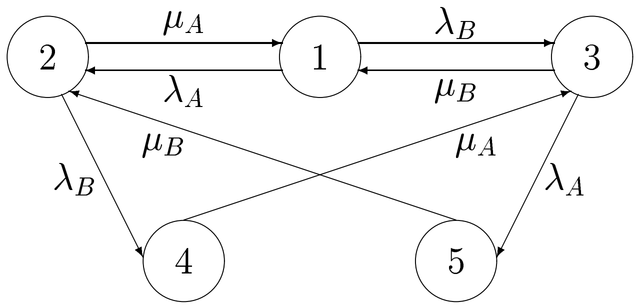
[1] Note, here the two equations are in fact the same. There will always be a redundant equation in the global balance equations which is why we need the additional information of the normalisation condition.

[2] i.e. rearranging the matrix so that the rows are columns and vice versa.

[3] Under DICE type `setup maple7` before typing `maple` or `xmaple`

Example: Consider the two-processor version of the system introduced in the previous section. We denote the processors A and B, respectively. We assume that the processors have different timing characteristics, the private memory access of A being governed by an exponential distribution with parameter λ_A, the common memory access of B being governed by an exponential distribution with parameter μ_B, etc. Now the state space becomes:

1. A and B both executing in their private memories;
2. B executing in private memory, and A accessing common memory;
3. A executing in private memory, and B accessing common memory;
4. A accessing common memory, B waiting for common memory;
5. B accessing common memory, A waiting for common memory;



$$\mathbf{Q} = \begin{pmatrix} -(\lambda_A+\lambda_B) & \lambda_A & \lambda_B & 0 & 0 \\ \mu_A & -(\mu_A+\lambda_B) & 0 & \lambda_B & 0 \\ \mu_B & 0 & -(\mu_B+\lambda_A) & 0 & \lambda_A \\ 0 & 0 & \mu_A & -\mu_A & 0 \\ 0 & \mu_B & 0 & 0 & -\mu_B \end{pmatrix}$$

This matrix is still small enough to be solved by hand but for the purposes of illustration we solve it using `maple`. The transcript of the `maple` session is given at the end of this note. After appropriate transformations the modified generator matrix is as follows:

$$\mathbf{Q}_n^T = \begin{pmatrix} -(\lambda_A+\lambda_B) & \mu_A & \mu_B & 0 & 0 \\ \lambda_A & -(\mu_A+\lambda_B) & 0 & 0 & \mu_B \\ \lambda_B & 0 & -(\mu_B+\lambda_A) & \mu_A & 0 \\ 0 & \lambda_B & 0 & -\mu_A & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

If we choose the following values for the parameters:

$$\lambda_A = 0.05 \qquad \lambda_B := 0.1 \qquad \mu_A = 0.02 \qquad \mu_B = 0.05$$

solving the matrix equation, and rounding figures to 4 significant figures, we obtain:

$$\boldsymbol{\pi} = (0.0693, 0.0990, 0.1683, 0.4951, 0.1683)$$

3.4 Derivation of Performance Measures

Roughly speaking, there are three ways in which performance measures can be derived from the steady state distribution of a Markov process. These different methods can be thought of as corresponding to different types of measure:

- state-based measures, e.g. utilisation;
- rate-based measures, e.g. throughput;
- other measures which fall outside the above categories, e.g. response time.

State-based measures are those which clearly correspond to the probability that the model is in a state, or a subset of states, which satisfy some condition. For example, utilisation will correspond to those states where a resource is in use. If we consider the example above, the utilisation of the common memory, U_{mem}, is the total probability that the model is in one of the states in which the common memory is in use:

$$U_{mem} = \boldsymbol{\pi_2} + \boldsymbol{\pi_3} + \boldsymbol{\pi_4} + \boldsymbol{\pi_5} = 93.07\%$$

Other examples of state-based measures are idle time, or the number of jobs in a system. Some measures such as the number of jobs will involve a weighted sum of steady state probabilities, weighted by the appropriate value. For example, if we consider jobs waiting for the common memory to be queued in that subsystem, then the average number of jobs in the common memory, N_{mem}, is:

$$N_{mem} = (1 \times \boldsymbol{\pi_2}) + (1 \times \boldsymbol{\pi_3}) + (2 \times \boldsymbol{\pi_4}) + (2 \times \boldsymbol{\pi_5}) = 1.594$$

Rate-based measures are those which correspond to the predicted rate at which some event occurs. This will be the product of the rate of the event, and the probability that the event is enabled, i.e. the probability of being in one of the states from which the event can occur. Thus, to calculate the throughput of the common memory, the average number of accesses from either processor which it processes in unit time, X_{mem}, we use

$$X_{mem} = (\mu_A \times (\boldsymbol{\pi_2} + \boldsymbol{\pi_4})) + (\mu_B \times (\boldsymbol{\pi_3} + \boldsymbol{\pi_5})) = 0.0287$$

or, approximately one access every 35 milliseconds.

Finally, the other category of measures are those which do not readily fit into either of the above categories. In these cases, we usually use one of the operational laws to derive the information we need, based on values that we have obtained from solution of the model. Thus, applying Little's law to the common memory we see that

$$W_{mem} = N_{mem}/X_{mem} = 1.594/0.0287 = 55.54 \text{ milliseconds}$$

3.5 Assumptions

The most fundamental assumption we have made is the **stochastic hypothesis**: *"The behaviour of a real system during a given period of time is characterised by the probability distributions of a stochastic process."* Although it can never be conclusively proved, evidence suggests that this hypothesis is true. In contrast, the assumption that all delays and inter-event times are exponentially distributed, is less likely to fit with observations of real systems. We make this assumption nevertheless because of the nice mathematical properties of the exponential distribution, and because the exponential distribution is the only one we can use if we wish to assume that the stochastic process characterising our system is a Markov process. The Markov assumption that future behaviour of the system is only dependent on its current state, not on its past history *is* a reasonable assumption for computer and communication systems, if we choose our states carefully. We assume that the Markov process is finite, time homogeneous and irreducible: necessary conditions for a steady state distribution to exist, and coincide with the long term probability distribution.

3.6 Transcript of maple session

```
> with(linalg):
>
> lambdaA := 0.05;
> muA := 0.02;
> lambdaB := 0.1;
> muB := 0.05;
>
> size := 5;
>
> Q := array(sparse,1..size,1..size);
> e := array(sparse,1..size);
>
> Q[1,2] := lambdaA;
>
> Q[1,3] := lambdaB;
>
> Q[2,1] := muA;
>
> Q[2,4] := lambdaB;
>
> Q[3,1] := muB;
>
> Q[3,5] := lambdaA;
>
> Q[4,3] := muA;
>
> Q[5,2] := muB;
>
> for i to size do
>     s := 0:
>     for j to size do
>         s := s + Q[i,j]
>     od;
>     Q[i,i] := -s
> od;
>
> Qt := transpose(Q);
>
> for i to size do Qt[size,i] := 1 od;
>
> e[size] := 1;
>
> p := linsolve(Qt,e);
```