Switch Fabrics

Switching Technology S38.165
http://www.netlab.hut.fi/opetus/s38165

Switch fabrics

- Multipoint switching
- Self-routing networks
- Sorting networks
- **Fabric implementation technologies**
- Fault tolerance and reliability
Fabric implementation technologies

- Time division fabrics
  - Shared media
  - Shared memory
- Space division fabrics
  - Crossbar
  - Multi-stage constructions
- Buffering techniques

Buffering alternatives

- Input buffering
- Output buffering
- Central buffering
- Combinations
  - input-output buffering
  - central-output buffering
Input buffering

Buffer memories at the input interfaces

- **Pros**
  - required memory access speed
    - in FIFO and dual-port RAM solutions equal to incoming line rate
    - in one-port RAM solutions twice the incoming line rate
  - Speed of switch fabric
    - multi-stages and crossbars operate at input wire speed
    - shared media fabrics operate at the aggregate speed of inputs
  - low cost solution (due to low memory speed)

- **Cons**
  - FIFO type of buffering => HOL problem
  - buffer size may be large (due to HOL)
  - HOL avoided by having a buffer for each output at each input
Output buffering

Buffer memories at the output interfaces

Output buffering (cont.)

- **Pros**
  - better throughput/delay performance than in input buffered systems
  - no HOL problem

- **Cons**
  - access speed of buffer memory
    - in FIFO and dual-port RAM solutions $N$ times the incoming line rate
    - in one-port RAM solutions $N+1$ times the incoming line rate
  - high cost due to high memory speed requirement
  - switch fabric operates at the aggregate speed of inputs ($N \times$ wire speed)
Central buffering

Buffer memory located between two switch fabrics
- shared by all inputs/outputs
- virtual buffer for each input or output

Central buffering (cont.)

• Pros
  • smaller buffer size requirement and lower average delay than in input or output buffering
  • HOL problem can be avoided

• Cons
  • speed of buffer memory
    - in dual-port RAM solutions larger than \( N \) times the incoming line rate
    - in one-port RAM solutions larger than \( 2 \times N \) times the incoming line rate
  • speed of switch fabric \( N \times \) wire speed
  • complicated buffer control
  • high cost due to high memory speed requirement and control complexity
Input-output buffering

Input-output buffering common in QoS aware switches/routers
- inputs implement output specific buffers to avoid HOL
- outputs implement dedicated buffers for different traffic classes
- combined buffering distributes buffering complexity between inputs and outputs

Input-central buffering

Input-central buffering used in QoS aware switches/routers
- inputs implement output specific buffers to avoid HOL
- central buffer implements dedicated buffers for different traffic classes for each output
Summary of buffering techniques

<table>
<thead>
<tr>
<th>Buffering principle</th>
<th>Memory space</th>
<th>Memory speed</th>
<th>Memory control</th>
<th>Queueing delay</th>
<th>Multi-casting capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input buffering</td>
<td>high</td>
<td>slow (~input rate)</td>
<td>simple</td>
<td>longest (due to HOL)</td>
<td>extra logic needed</td>
</tr>
<tr>
<td>Output buffering</td>
<td>medium</td>
<td>fast (~N x input rate)</td>
<td>simple</td>
<td>medium</td>
<td>supported</td>
</tr>
<tr>
<td>Central buffering</td>
<td>low</td>
<td>fast (~N x input rate)</td>
<td>complicated</td>
<td>shortest</td>
<td>supported but complex</td>
</tr>
</tbody>
</table>

Priorities and buffering

- Separate buffer for each traffic class
- A scheduler needed to control transmission data
  - highest priority served first
  - longest queue served first
  - minimization of lost packets/cells
- Priority given to high quality traffic
  - low delay and delay variation traffic
  - low loss rate traffic
  - best customer traffic
- Scheduling principles
  - round robin
  - weighted round robin
  - fair queuing
  - weighted fair queuing
  - etc.
Basic memory types for buffering

- FIFO (First-In-First-Out)
- RAM (Random Access Memory)
- Dual-port RAM

Basic memory types for buffering (cont.)
Switch fabrics

- Multipoint switching
- Self-routing networks
- Sorting networks
- Fabric implementation technologies
- **Fault tolerance and reliability**

Fault tolerance and reliability

- Definitions
- Fault tolerance of switching systems
- Modeling of tolerance and reliability
Definitions

- **Failure, malfunction** - is deviation from the intended/specified performance of a system
- **Fault** - is such a state of a device or a program which can lead to a failure
- **Error** - is an incorrect response of a program or module. An error is an indication that the module in question may be faulty, the module has received wrong input or it has been misused. An error can lead to a failure if the system is not tolerant to this sort of an error. A fault can exist without any error taking place.

Fault tolerance

- **Fault tolerance** is the ability of a system to continue its intended performance in spite of a fault or faults
- **A switching system** is an example of a fault tolerant system
- Fault tolerance always requires redundancy of some sort
Categorization of faults

- **Duration based**
  - **permanent** or stuck-at (stuck at zero or stuck at one)
  - **intermittent** - fault requires repair actions, but its impact is not always observable
  - **transient** - fault can be observed for a short period of time and disappears without repair
- **Observable** or **latent** (hidden)
- Based on the **scope** of the impact (serious - less serious)

Graceful degradation

- **Capability of a system to continue its functions under one or more faults, but on a reduced level of performance**
- **For example**
  - in some **RAID** (Redundant Array Inexpensive Disks) configurations, write speed drops in case of a disk fault, but continues on a lower level of performance even while the fault has not been repaired
Reliability and availability

- **Reliability** $R(t)$ - probability that a system does not fail within time $t$ under the condition that it was functioning correctly at $t = 0$
  - for all known man-made systems $R(t) \to 0$ when $t \to \infty$
- **Availability** $A(t)$ - probability that a system will function correctly at time $t$
  - for a system that can be repaired $A(t)$ approaches some value asymptotically during the useful lifetime of the system

Repairable system

- **Maintainability** $M(t)$ - probability that a system is returned to its correct functioning state during time $t$ under the condition that it was faulty at time $t = 0$
MTTF, MTTR and MTBF

- **MTTF (Mean-Time-To-Failure)** - expected value of the time duration from the present to the next failure
- **MTTR (Mean-Time-To-Repair)** - expected value of the time duration from a fault until the system has been restored into a correct functioning state
- **MTBF (Mean-Time-Between-Failures)** - expected value of the time duration from occurrence of a fault until the next occurrence of a fault
  - \[ MTBF = MTTR + MTTR \]

High availability of a switching system

- High availability of a switching system is obtained by maintenance software

- **Supervision**
  - Detection of errors and faults
  - In a unit under normal working load
  - HW implementation fast
  - SW implementation as detection delay

- **Alarm system**
  - Fault analysis and pinpointing
  - Often a rule based system

- **Recovery**
  - Recovery - elimination of faults
  - Utilizes
    - redundancy
    - switch-overs
      - active ↔ standby
    - restarts
    - a single program
    - a preprocessor
    - a single main processor
    - whole system
    - fall back to previous SW package

- **Diagnostics**
  - Fault location
  - In a unit temporarily without normal load

**Maintenance software is one of the most important software sub-systems in a switching system in parallel with call/connection control and charging**
Main types of redundancy

- **Hardware redundancy**
  - duplication (1+1) - need for “self-checking”-recovery blocks that detect their own faults
  - $n+r$-principle ($n$ active units and $r$ standby units)

- **Software redundancy**
  - required always in telecom systems

- **Information redundancy**
  - parity bits, block codes, etc.

- **Time redundancy**
  - delayed re-execution of transactions

Modeling of reliability

- Combinatorial models
- Markov analysis
- Other modeling techniques *(not covered here)*
  - Fault tree analysis
  - Reliability block diagrams
  - Monte Carlo simulation
### Combinatorial reliability

- A **serial system** $S$ functions if and only if all its parts $S_i (1 \leq i \leq n)$ function

  $$R_s = \prod_{i=1}^{n} R_i \text{ and } F_s = (1 - R_s)$$

- Failures in sub-systems are supposed to be independent

- A **parallel** (replicated) system fails if all its sub-systems fail

  $$F_s = \prod_{i=1}^{n} (1 - R_i) \text{ and } R_s = 1 - F_s = 1 - \prod_{i=1}^{n} (1 - R_i)$$

- Reliability of a duplicated system ($R_i = R$) is

  $$R_s = 1 - (1 - R)^2$$

### Combinatorial reliability example 1

- Calculate reliability $R_s$ and failure probability $F_s$ of system $S$ given that failures in sub-systems $S_i$ are independent and for some time interval it holds that $R_1 = 0.90$, $R_2 = 0.95$ and $R_3 = R_4 = 0.80$

  $$R_s = \prod_{i=1}^{n} R_i = R_1 \times R_2 \times R_{3-4}$$

  $$R_{3-4} = 1 - \prod_{i=1}^{n} (1 - R_i) = 1 - (1 - R_3)(1 - R_4)$$

  $$R_s = R_1 \times R_2 \times [1 - (1 - R_3)(1 - R_4)]$$

  $$F_s = 1 - R_s = 1 - R_1 \times R_2 \times [1 - (1 - R_3)(1 - R_4)]$$

  $$R_s = 0.82 \text{ and } F_s = 0.18$$
Combinatorial reliability (cont.)

- A load sharing system functions if \( m \) of the total of \( n \) sub-systems function
- If failures in sub-systems \( S_i \) are independent then probability that the system fails is
  \[
P(\text{fails}) = P(k< m)
  \]
  and probability that it functions is
  \[
P(\text{functioning}) = P(k \geq m) = 1 - P(k< m)
  \]
  where \( k \) is the number of functioning sub-systems

\[
P(k \geq m) = \sum_{i=m}^{n} P(k=i) \quad \text{and} \quad P(k< m) = \sum_{i=0}^{m-1} P(k=i)
\]

Combinatorial reliability example 2

- As an example, suppose we have a system having \( m=2 \) and \( n=4 \) and each of the four sub-systems have a different \( R \), i.e. \( R_1, R_2, R_3 \) and \( R_4 \), and failures in sub-systems \( S_i \) are independent

- Probability that the system fails is
  \[
P(\text{fails}) = P(k<2) = \sum_{i=0}^{1} P(k=i) = P(k=0) + P(k=1)
  \]

- \( P(k=0) \) and \( P(k=1) \) can be derived to be
  \[
P(k=0) = (1- R_1)(1- R_2)(1- R_3)(1- R_4)
P(k=1) = R_1(1- R_2)(1- R_3)(1- R_4) + (1- R_1)R_2(1- R_3)(1- R_4) + (1- R_1)(1- R_2)R_3(1- R_4) + (1- R_1)(1- R_2)(1- R_3)R_4
  \]

- If \( R_1=0.9 \), \( R_2=0.95 \), \( R_3=0.85 \), and \( R_4=0.8 \) then
  \[
  R_s = 0.994 \quad \text{and} \quad F_s = 0.0058
  \]

© P. Raatikainen
Switching Technology / 2003
7 - 31

© P. Raatikainen
Switching Technology / 2003
7 - 32
Combinatorial reliability (cont.)

- If failures in sub-systems $S_i$ of an $m/n$ system are independent and $R_i = R$ for all $i \in [1,n]$ then the system is a Bernoulli system and binomial distribution applies

$$R_s = \sum_{k=0}^{n} \binom{n}{k} R^k (1-R)^{n-k}$$

- For a system of $m/n = 2/3$

$$R_{2/3} = \sum_{k=2}^{3} \frac{3!}{k!(3-k)!} R^k (1-R)^{3-k} = 3R^2 - 2R^3$$

If for example $R = 0.9 \Rightarrow R_{2/3} = 0.972$

Computing MTTF

- $MTTF = \int_0^\infty R(t) dt$ - valid for any reliability distribution

- Single component with a constant failure rate (CFR) $\lambda$
  - $R(t) = e^{-\lambda t}$
  - $MTTF = \frac{1}{\lambda}$

- Serial systems with $n$ CFR components
  - $R_s(t) = R_1(t) \times R_2(t) \times \ldots \times R_n(t) = e^{-(\lambda_1 + \lambda_2 + \ldots + \lambda_n)t} = e^{-\lambda_s t}$
  - $\lambda_s = \lambda_1 + \lambda_2 + \ldots + \lambda_n$
  - $MTTF_s = \frac{1}{\lambda_s}$
  - $1/MTTF_s = 1/MTTF_1 + 1/MTTF_2 + \ldots + 1/MTTF_n$
Telecom exchange reliability from subscriber’s point of view

- Line-card
- Subscriber module control
- Subscriber call control
- Centralized functions
- CCS7 signaling processors
  - \((n-1)/n\) operational processors for call setup
  - chosen processor functions during a call
- Exchange terminal

Premature release requirement \( \mathbf{P} \leq 2 \times 10^{-5} \) applied

---

### Failure intensity

- Unit of failure intensity \( \lambda \) is defined to be
  \[ \lambda = \text{fit} = \text{number of faults} / 10^9 \text{ h} \]
- Failure intensities for replaceable plug-in-units varies in the range 0.1 - 10 kfit

- Example:
  - if failure intensity of a line-card in an exchange is 2 kfit, what is its MTTF?

  \[
  \text{MTTF} = \frac{1/\lambda}{2000} = \frac{10^9 \text{ h}}{2000} = \frac{1000000 \text{ h}}{2 \times 24 \times 360} = 58 \text{ years}
  \]
Reliability modeling using Markov chains

Markov chains

• A system is modeled as a set of states of transitions
• Each state corresponds to fulfillment of a set of conditions and each transition corresponds to an event in a system that changes from one state to another

• By using this method it is possible to find reliability behavior of a complex system having a number of states and non-independent failure modes

Markov chain modeling

• A set of states of transitions leads to a group of linear differential equations
• For a given modeling goal it is essential to choose a minimal set of states for equations to be easily solved
• By setting the derivatives of the probabilities to zero an asymptotic state is obtained if such exists

\[ \lambda = \text{failure intensity} \]
\[ \mu = \text{repair intensity (repair time is exponentially distributed)} \]
\[ P_i = \text{probability of state } i, \text{ e.g. } P_0 = R(t) \text{ and } P_1 = F(t). \]
Markov chain modeling (cont.)

- Probabilities ($\pi$) of the states and transition rates ($\lambda_{ij}$) between the states are tied together with the following formula

$$\pi \Lambda = 0$$

where

$$\pi = [\pi_1, \pi_2, \ldots, \pi_n]$$

$$\Lambda = \begin{bmatrix}
-(\lambda_{12} + \lambda_{13}) & \lambda_{12} & \lambda_{13} & \cdots \\
\lambda_{21} & -(\lambda_{21} + \lambda_{23}) & \lambda_{23} & \cdots \\
\lambda_{31} & \lambda_{32} & -(\lambda_{31} + \lambda_{32}) & \cdots \\
& & & \ddots & \\
& & & & & & \\
\end{bmatrix}$$

Example

$$\Lambda = \begin{bmatrix}
-(\lambda_{12} + \lambda_{13}) & \lambda_{12} \\
\lambda_{21} & -(\lambda_{21} + \lambda_{23}) \\
\lambda_{31} & \lambda_{32} & -(\lambda_{31} + \lambda_{32}) \\
& & & \ddots & \\
& & & & & & \\
\end{bmatrix}$$

$$\pi \Lambda = 0 \quad \text{and} \quad \pi = [\pi_1, \pi_2, \ldots, \pi_n]$$

$$\begin{cases}
-(\lambda_{12} + \lambda_{13})\pi_1 + \lambda_{12}\pi_2 + \lambda_{13}\pi_3 = 0 \\
\lambda_{21}\pi_1 - (\lambda_{21} + \lambda_{23})\pi_2 + \lambda_{23}\pi_3 = 0 \\
\lambda_{31}\pi_1 + \lambda_{32}\pi_2 - (\lambda_{31} + \lambda_{32})\pi_3 = 0
\end{cases}$$
Birth-death process

Birth-death process is a special case of continuous-time Markov chain, which models the size of population that increases by 1 (birth) or decreases by one (death).

Balance equations:
- State $S_0$ \[ \lambda_0 \pi_0 = \lambda_1 \pi_1 \] \[ \Rightarrow \pi_i = \frac{\lambda_i}{\mu_i} \pi_0 \]
- State $S_1$ \[ (\lambda_1 + \mu_1)\pi_1 = \lambda_0 \pi_0 + \lambda_2 \pi_2 \] \[ \Rightarrow \pi_1 = \frac{\lambda_0}{\mu_1} \pi_0 \]
- State $S_k$ \[ (\lambda_{k-1} + \mu_{k-1})\pi_{k-1} = \lambda_{k-2} \pi_{k-2} + \lambda_k \pi_k \] \[ \Rightarrow \pi_k = \frac{\lambda_k}{\mu_k} \frac{\lambda_{k-2}}{\mu_{k-2}} \frac{\lambda_{k-1}}{\mu_{k-1}} \pi_0 \] (for $k = 1, 2, 3, \ldots$)

Substituting these expressions for \( \pi_k \) into \( \sum \pi_i = 1 \) yields

\[ \pi_0 + \sum_{i=1}^{\infty} \frac{\lambda_{i-1} \lambda_i \lambda_{i+1}}{\mu_{i-1} \mu_i \mu_{i+1}} \pi_0 = 1 \]
\[ \Rightarrow \pi_0 \left[ 1 + \sum_{i=1}^{\infty} \frac{\lambda_{i-1} \lambda_i \lambda_{i+1}}{\mu_{i-1} \mu_i \mu_{i+1}} \right] = 1 \]
\[ \Rightarrow \frac{1}{\pi_0} = 1 + \sum_{i=1}^{\infty} \frac{\lambda_{i-1} \lambda_i \lambda_{i+1}}{\mu_{i-1} \mu_i \mu_{i+1}} \]
\[ \Rightarrow \pi_0 = \frac{\lambda_{i-1} \lambda_i \lambda_{i+1}}{1 + \sum_{i=1}^{\infty} \frac{\lambda_{i-1} \lambda_i \lambda_{i+1}}{\mu_{i-1} \mu_i \mu_{i+1}}} \] (for $k = 1, 2, 3, \ldots$)
A switching system has two control computers, one on-line and one standby. The time interval between computer failures is exponentially distributed with mean $t_f$. In case of a failure, the standby computer replaces the failed one. A single repair facility exist and repair times are exponentially distributed with mean $t_r$. What fraction of time the system is out of use, i.e., both computers having failed?

The problem can be solved by using a three state birth-death model.

![Birth-death process diagram]

If $t_r/t_f = 10$, i.e. the average repair time is 10 % of the average time between failures, then $\pi_0 = 0.009009$ and both computer will be out of service 0.9 % of the time.
Additional reading of Markov chain modeling

Switching Technology S38.165
http://www.netlab.hut.fi/opetus/s38165

Markov chain modeling

A continuous-time Markov Chain is a stochastic process \( \{ X(t): t \geq 0 \} \)

- \( X(t) \) can have values in \( S=\{0,1,2,3,\ldots\} \)
- Each time the process enters a state \( i \), the amount of time it spends in that state before making a transition to another state has an exponential distribution with mean \( 1/\lambda_i \)
- When leaving state \( i \), the process moves to a state \( j \) with probability \( p_{ij} \) where \( p_{ii}=0 \)
- The next state to be visited after \( i \) is independent of the length of time spend in state \( i \)
Markov chain modeling (cont.)

Transition probabilities

\[ p_{ij}(t) = P\{X(t+s) = j | X(s) = i\} \]

Continuous at \( t=0 \), with

\[ \lim_{t \to 0} p_{ij}(t) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \]

Transition matrix is a function of time

\[ P(t) = \begin{bmatrix} p_{11}(t) & p_{12}(t) & \cdots \\ p_{21}(t) & \vdots & \ddots \\ \vdots & \ddots & \ddots \end{bmatrix} \]

Markov chain modeling (cont.)

Transition intensity:

\[ \lambda_i(t) = \frac{d}{dt} p_{ij}(0) \] (rate at which the process leaves state \( j \) when it is in state \( i \))

\[ \lambda_j(t) = \frac{d}{dt} p_{ij}(0) = \lambda_i p_{ij} \] (transition rate into state \( j \) when the process is in state \( i \))

The process, starting in state \( i \), spends an amount of time in that state having exponential distribution with rate \( \lambda_i \). It then moves to state \( j \) with probability

\[ p_{ij} = \frac{\lambda_j}{\lambda_i} \quad \forall i,j \]

\[ \sum_{j=1}^{n} p_{ij} = \sum_{j=1}^{n} \lambda_j \left( \frac{1}{\lambda_i} \right) = 1 \quad \Rightarrow \quad \lambda_i = \sum_{j=1}^{n} \lambda_{ij} \]
Markov chain modeling (cont.)

Chapman-Kolmogorov equations:
\[ p_{ij}(t+s) = \sum_{k \in S} p_{ik}(t)p_{kj}(s) \quad \forall i,j \in S \]
\[ \forall s,t \geq 0 \]

Since \( p(t) \) is a continuous function
\[ p_{ij}(\Delta t) = p_{ij}(0) + \frac{d}{dt} p_{ij}(0)\Delta t + o(\Delta t^2) \]

We have defined \( \lambda_{ij}(t) = \frac{d}{dt} p_{ij}(0) \)

For \( \neq j \):
\[ p_{ij}(\Delta t) = p_{ij}(0) + \lambda_{ij}\Delta t + o(\Delta t^2) = \lambda_{ij}\Delta t \quad \text{(for small } \Delta t) \]

For \( i=j \):
\[ p_{ii}(\Delta t) = p_{ii}(0) + \lambda_{ii}\Delta t + o(\Delta t^2) = 1 + \lambda_{ii}\Delta t \quad \text{(for small } \Delta t) \]

From Chapman-Kolmogorov equations:
\[ p_{ij}(t+\Delta t) = \sum_{k} p_{ik}(t)p_{kj}(\Delta t) = p_{ij}(t)p_{ij}(\Delta t) + \sum_{k\neq j} p_{ik}(t)p_{kj}(\Delta t) + \sum_{i\neq j} p_{ij}(t)\Delta t + o(\Delta t^2) \]

\[ p_{ij}(t+\Delta t) = p_{ij}(t) + \sum_{i\neq j} p_{ij}(t)\lambda_{ij} \Delta t + \sum_{i\neq j} p_{ij}(t)\lambda_{ij} \Delta t + o(\Delta t^2) \]

\[ \frac{p_{ij}(t+\Delta t) - p_{ij}(t)}{\Delta t} = \sum_{i\neq j} p_{ij}(t)\lambda_{ij} + \sum_{i\neq j} p_{ij}(t)\lambda_{ij} \Delta t + o(\Delta t^2) \]

Taking the limit as \( \Delta t \to 0 \)
\[ \frac{d}{dt} p_{ij}(t) = \sum_{i\neq j} p_{ij}(t)\lambda_{ij} \quad \forall i,j \]
Markov chain modeling (cont.)

The process is described by the system of differential equations:

\[
\frac{d}{dt} p_{ij}(t) = \sum_k p_{ik}(t) \lambda_{kj} \quad \forall i,j
\]

which can be given in the form

\[
\frac{d}{dt} P(t) = P(t) \Lambda \quad \forall i,j \quad \sum_j p_{ij}(t) = 1 \quad \forall i, t
\]

\[
\frac{d}{dt} \sum_j p_{ij}(t) = \frac{d}{dt} (1) = 0
\]

\[
\sum_j \lambda_{ij} = 0 \quad \text{The sum of each row of } \Lambda \text{ is zero!}
\]

Markov chain modeling (cont.)

Example

\[
\Lambda = \begin{bmatrix}
-\lambda_{42} + \lambda_{43} & \lambda_{42} & \lambda_{43} \\
\lambda_{21} & -\lambda_{21} + \lambda_{23} & \lambda_{23} \\
\lambda_{31} & \lambda_{32} & -\lambda_{31} + \lambda_{33}
\end{bmatrix}
\]

The sum of each row of \( \Lambda \) must be zero!
Markov chain modeling (cont.)

Steady state probabilities

\[ \lim_{t \to \infty} p_{ij}(t) = \pi_j \quad \text{(Independent of initial state } i) \]

Must be non-negative and must satisfy \( \sum_{i=1}^{n} \pi_i = 1 \)

In case of continuous-time Markov chains balance equation used to determine \( \pi \).
For each state \( i \), the rate at which the system leaves the state must equal to the rate at which the system enters the state

\[ \lambda_i \pi_i = \lambda_{ij} \pi_j + \lambda_{ik} \pi_k \]

Markov chain modeling (cont.)

Balance equation

\[ \left( \sum_{j \neq i} \lambda_{ij} \right) \pi_i = \sum_{k \neq i} \lambda_{ik} \pi_k \quad \forall i \]

Steady state distribution is computed by solving this system of equations

\[ \left( \sum_{j \neq i} \lambda_{ij} \right) \pi_i = \sum_{k \neq i} \lambda_{ik} \pi_k \quad \forall i \]

\[ \sum_{i=1}^{n} \pi_i = 1 \]
Markov chain modeling (cont.)

An alternative derivation of the steady-state conditions begins with the differential equation describing the process:

$$\frac{d}{dt} p_i(t) = \sum_k p_k(t) \lambda_{ij} \quad \forall i, j$$

Suppose that we take the limit of each side as \( t \to \infty \)

$$\Rightarrow \lim_{t \to \infty} \frac{d}{dt} p_i(t) = \lim_{t \to \infty} \sum_k p_k(t) \lambda_{ij}$$

$$\Rightarrow \frac{d}{dt} \lim_{t \to \infty} p_i(t) = \sum_k \lim_{t \to \infty} p_k(t) \lambda_{ij}$$

$$\Rightarrow \sum_k \pi_k \lambda_{ij} = 0 \quad \text{i.e. } \pi \Lambda = 0$$

Markov chain modeling (cont.)

Example

$$\Lambda = \begin{bmatrix} -\lambda_{12} - \lambda_{13} & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & -(\lambda_{21} + \lambda_{23}) & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & -(\lambda_{31} + \lambda_{33}) \end{bmatrix}$$

$$\pi \Lambda = 0 \quad \text{and} \quad \pi = [\pi_1 \quad \pi_2 \quad \ldots \quad \pi_s]$$

$$\begin{cases} - (\lambda_{12} + \lambda_{13}) \pi_1 + \lambda_{21} \pi_2 + \lambda_{31} \pi_3 = 0 \\ \lambda_{12} \pi_1 - (\lambda_{21} + \lambda_{23}) \pi_2 + \lambda_{32} \pi_3 = 0 \\ \lambda_{13} \pi_1 + \lambda_{32} \pi_2 - (\lambda_{31} + \lambda_{33}) \pi_3 = 0 \end{cases}$$