Leistungsanalyse von Rechnersystemen

Introduction to Queuing Theory

Nöthnitzer Straße 46
Raum 1026
Tel. +49 351 - 463 - 35048

Holger Brunst (holger.brunst@tu-dresden.de)
Matthias S. Mueller (matthias.mueller@tu-dresden.de)

Summary of Previous Lecture

Performance Simulation and Prediction

Nöthnitzer Straße 46
Raum 1026
Tel. +49 351 - 463 - 35048

Holger Brunst (holger.brunst@tu-dresden.de)
Matthias S. Mueller (matthias.mueller@tu-dresden.de)
Simulation Code are Complex

- LOC for different simulation codes:

- Developing a simulation package is a complex software engineering task with the additional complexity that your first assumptions, e.g. the modeling might be wrong.
- There are many different simulation methods, each with its own complexity.
- Even simple methods might be difficult to implement, remember the stories from real life:
  - Loosing particles
  - Inappropriate integrators

Random Number Generators (RNG)

- Pseudo-Random (should generate reproducible sequence of pseudo-random numbers):
  \[ x_i \]

- Should have well defined distribution (uniformly distributed)
- Fast
- High quality:
  - No correlation: \[ \langle x_i x_{i+n} \rangle = \langle x_i \rangle^2 \]
  - Long sequence length
### Congruential Generators

- Iterative generation of pseudo-random numbers:
  \[ x_{i+1} = (cx_i) \mod p \]

- Condition for high quality numbers:
  - Theory:
    - \( p \) is prime
    - \( P \) smallest number with: \( c^{p-1} \mod p = 1 \)
  - Practice: use literature
    - “Minimal Standard”, Park and Miller
      \[ p = 2^{31} - 1 \]
      \[ c = 7^5 = 16807 \]

### Kirkpatrick-Stoll Generator

- Use 32 lagged Fibonacci Generators in parallel, one for each bit
- Generation of random numbers:
  \[ x_0 = (x_{103} + x_{250}) \mod 2 \]

- Problems:
  - First 250 numbers have to be generated elsewhere
  - Different sequences have to be uncorrelated

- Advantage:
  - Very long sequence
  - Fast (Bit operations/ additions MOD 2 is XOR)
Tests for RNGs

- Check distribution
- Check mean value
- Check if mean value of all bits is 0.5
- Check n-cube correlations
- Check correlations
  \[ \langle x_i x_{i+n} \rangle = \langle x_i \rangle^2 \]
- Power distribution in frequency space (white noise)
- Check if partial sums of a sequence have Gaussian distribution (Chi-square test)

N-cube Correlations

- Choose n-tupels from sequence: \((x_i, x_{i+1}, \ldots, x_{i+n})\), \((x_{i+1}, x_{i+2}, \ldots, x_{i+n+1})\),
- Interpret them as points in n-dimensional space
- Plot them (example for \(p=31, c=3\) and \(p=31, c=11\)):
Recommendations for using RNGs

- Do not subdivide one stream (n-cube correlations)
- Consider the periodicity: use non-overlapping streams
- Seed once in your application run, not every time you enter a specific subtask
- Do not use random seeds, or at least document them
- **Reproduce your results with different seeds AND different RNGs!**

There are famous examples that a “physical property, e.g. in phase transition simulations” was depending on the RNG used.
If the facts don’t fit the theory, change the facts.

Albert Einstein

Outline

- Motivation
- Queuing/Kendall notation
- Queuing in daily life
- Exponential distribution and its memoryless property
- Little’s law
- Stochastic processes, birth-death process
- M|M|1 queuing model
Motivation

- **Sharing of system resources** in computer systems:
  - CPU, Disk, Network, etc.
- Generally, only one job can use the resource at any time
- All other jobs using the same resource **wait in queues**
- Queuing or queuing theory helps in determining the time that jobs spend in various system queues.
- These times can be combined to predict the response time of jobs

Queuing Notation

- Imagine yourself at a supermarket checkout
- The checkout has a number of open cash points
- Usually, the cash points are busy and arriving customers have to wait
- In queuing theory terms you would be called “**customer**” or “**job**”
- In order to analyze such systems, the following system characteristics should be specified:
  1. Arrival Process
  2. Service Time Distribution
  3. Number of Servers
  4. System Capacity
  5. Population Size
  6. Service Discipline
Queuing Notation

1. Arrival Process (Ankunftsprozess)
   - If customers arrive at $t_1, t_2, \ldots, t_j$, the random variables $\tau_j = t_j - t_{j-1}$ are called interarrival times (Zwischenankunftszeiten).
   - General assumption: The $\tau_j$ form a sequence of independent and identically distributed (IID) random variables.
   - Most common arrival process is the Poisson Process which has exponentially distributed inter-arrival times.
   - Erlang- and hyper-exponential distributions are also used.

2. Service Time Distribution (Antwortzeitverteilung)
   - The time a customer spends at the service station e.g. the cash points.
   - This time is called the service time (Antwortzeit).
   - Commonly assumed to be IID random variables.
   - Exponential distribution is often used.

3. Number of Servers (Anzahl der Bedienstationen)
   - The number of service providing entities available to customers.
   - If in the same queuing system, servers are assumed to be:
     - Identical
     - Available to all customers.

4. System Capacity
   - The maximum number of customers who can stay in the system.
   - In most systems the capacity is finite.
   - However, if the number is large, infinite capacity is often assumed for simplicity.
   - The number includes both waiting and served customers.
Queuing Notation

5. Population Size
   - The total number of serviced customers
   - In most real systems the population is finite
   - If this size is large, once again, the size is assumed infinite for simplicity reasons

6. Service Discipline or Scheduling
   - The order in which customers are served:
     • First come first served (FCFS)
     • Last come first served (LCFS) with or without preemption (PR)
     • Round Robin (RR) with fixed size quantum
     • Processor Sharing (PS) if quantum size if small compared to average service time
     • Infinite Server (IS). System with fixed delay e.g. satellite link
     • Prioritized scheduling (PRIO)

Kendall Notation

These six parameters need to be specified in order to define a single queuing station

To compactly describe the queuing station in an unambiguous way, the so-called Kendall Notation is often used:

Arrivals | Services | Servers | Capacity | Population | Scheduling

- Arrivals ➔ customer arrival process
- Services ➔ customer service requirements
- Servers ➔ number of service providing entities
- Capacity ➔ maximum number of customers in queuing station
- Population ➔ size of the customer population
- Scheduling ➔ employed scheduling strategy

Population and Scheduling are often omitted i.e. assumed to be infinitely and FCFS
Kendall Notation

The specific values of the parameters, especially **Arrivals** and **Services**, are diverse. Some commonly used ones are:

- **M** (Markovian or Memory-less): whenever the interarrival or service times are (negative) exponentially distributed
- **G** (General): whenever the times involved may be arbitrarily distributed
- **D** (Deterministic): whenever the times involved are constant
- **E_r** (r-stage Erlang): whenever the times involved are distributed according to an r-stage Erlang distribution
- **H_r**: whenever the times involved are distributed according to an r-stage hyper-exponential distribution

Kendall Notation - Example

**M|G|2|8||LCFS** denotes a queuing station with:

- Negative exponentially distributed interarrival times
- Generally distributed service times
- 2 service providing entities
- Maximal 8 customers present
- No limitation on the total customer population
- LCFS scheduling strategy

Simple queuing stations as above can be used for many queuing phenomena in computer and communication systems.

However, just a single queue with single service entity is considered, only allowing performance evaluations of parts of a complex system.

Examples: Analysis of network access mechanisms, simple transmission links, or various disk and CPU scheduling mechanisms.
Queuing in Daily Life

- Coin-operated coffee machines
  - Service time, i.e., the time for preparing the coffee, is deterministic
  - Waiting time occurs due to the stochastic in the arrival process
  - Kendall notation: G|D|1

- Visiting a doctor with appointment
  - Arrival times of patients is deterministic (if their appointments are accurate)
  - However, one often experiences long waiting times due to the stochastic service times (time the doctor talks to or examines patients)
  - Kendall notation: D|G|1

- Visiting a doctor without appointment
  - Things become get even worse during “walk-in” consulting hours
  - Both arrival and service process obeys only general characteristics and the perceived waiting time increases
  - Kendall notation: G|G|1

Exponential (Markov) Distribution

- The (negative) exponential distribution is used extensively in queuing models
- It is the only continuous distribution with the so-called memoryless property which strongly simplifies the analysis:
  - Remembering the time since the last event does not help in predicting the time till the next event!
- Commonly used to model random durations, e.g.:
  - Duration of a phone call, Time between two phone calls
  - Duration of services, reparations, maintenance
  - Lifetime of radioactive atoms
  - Lifetime of parts, machines, technical equipment (without decline!)
Exponential Distribution

**Probability density function** (Dichtefunktion), short: pdf

\[ f(x; \lambda) = \begin{cases} \lambda e^{-\lambda x} , & x \geq 0, \\ 0 , & x < 0. \end{cases} \]

- Supported on interval \([0, \infty)\)
- \(\lambda > 0\) is a parameter of the distribution
- Often called rate parameter
- Probability of continuous random variable \(X\):
  \[ P(a \leq X \leq b) = \int_a^b f(x) \, dx \]

\[
\begin{align*}
\lambda &= 0.5 \\
\lambda &= 1.0 \\
\lambda &= 1.5
\end{align*}
\]

Exponential Distribution

**Cumulative distribution function** (Verteilungsfunktion), short CDF

\[ F(x, \lambda) = \begin{cases} 1 - e^{-\lambda x} , & x \geq 0, \\ 0 , & x < 0. \end{cases} \]

- Mean:
  \[ E[X] = \frac{1}{\lambda} \]
- Variance:
  \[ \text{Var}[X] = \frac{1}{\lambda^2} \]
Memoryless Property

- Stated earlier: Remembering the time since the last event does not help in predicting the time till the next event!
- Probability distribution of an exponentially distributed event $T$ to occur within time $t$:
  \[ F(T) = P(T < t) = 1 - e^{-\lambda t}, t \geq 0 \]
- We see an arrival event and start the clock at $t = 0$. The mean time to the next arrival event is $1/\lambda$.
- Suppose we do not see an arrival event until $t = x$. The distribution of the time remaining until the next arrival is:
  \[
P(T < x + t | T > x) = \frac{P(x < T < x + t)}{P(T > x)} = \frac{P(T < x + t) - P(T < x)}{P(T > x)} = \frac{(1 - e^{-\lambda(x+t)}) - (1 - e^{-\lambda x})}{e^{-\lambda x}} = 1 - e^{-\lambda t}
  \]

Memoryless Property

- A random variable $T$ is said to be memoryless if:
  \[ P(T < x + t | T > x) = P(T < t) \quad \forall \ x, t \geq 0 \]
- Common mistake:
  - (right) \quad P(T > 60 | T > 50) = P(T > 10)
  - (wrong) \quad P(T > 60 | T > 50) = P(T > 60)
  - The two events are not independent
- Check:
  - Give a real-life example whose lifetime can be modeled by a variable $T$ such that $P(T > s + t | T > s)$ goes down as $s$ goes up
  - Bus with exponentially distributed arrival times and $\lambda = 2/h$
    - Average waiting time?
    - Expected waiting time when already waiting for 15 minutes?
Little’s Law

- Named after John Little (MIT) who proved the law in 1961
- One of the most general laws in performance analysis
- Can be applied almost unconditionally to all queuing models and at many levels of abstraction
- Interesting point of notice: Long used before actually proved

Little’s Law basically relates the average number of jobs $N$ in queuing station to the average number of arrivals per time unit $\lambda$ and the average time $R$ spent in the queuing station

$$N = \lambda R$$

Little’s Law - Understanding

- Consider a queuing station as a black box
- On average $\lambda$ jobs arrive per time unit
- Upon its arrival, a job is either served or has to wait
- Denote $E[R]$ (residence time or response time) as the average time spend in the queuing system
- Denote average number of jobs in the queuing system as $E[N]
- Observe a single marked job which enters the system at $t=t_i$ leaves at $t=t_o$.
- On average $t_o - t_i$ will be equal to $E[R]$.
- While this particular job passes the system, other jobs have arrived
- Since on average $E[R]$ time units elapsed, their average number is $\lambda \times E[R]$
- This number must be equal to the previously defined $E[N]$ as every job could be the marked job. Thus:

$$E[N] = \lambda E[R]$$
Little’s Law - Remarks

- We assumed that the queue throughput $T$ equals the arrival rate $\lambda$.
- Always the case if system is not overloaded and infinite buffers.
- Otherwise customers will get lost and $E[N] = T E[R]$.
- The relationship expressed by Little’s law is valid independently of the form of the involved distributions.
- This law is valid independently of the scheduling discipline and the number of servers.
- $E[N]$ is easy to obtain and measures like $E[R]$ can be derived from it.
- Applies also to networks of queuing stations.

Stochastic Processes

- Analytical modeling uses several random variables but also **stochastic processes** which are sequences of random variables.
- Collection of random variables $\{X(t) \mid t \in T\}$, indexed by the parameter $t$ (usually time) which can take values of set $T$.
- Values that $X(t)$ assumes are called **states**. All possible states are called **state space** $I$.
- The state space and the time parameter can be discrete or continuous.
- Discrete-state stochastic processes are also called **chain**, often with $I = \{0,1,2,\ldots\}$.
Birth-Death Process

- Future states of the process are independent of the past and depend only on the present
- Special case of the continuous time Markov chain
- State transitions are restricted to neighboring states
- States are represented by integers. State $n$ can only change to state $n+1$ or state $n-1$
- Example: The number of jobs in a queue with a single server and individual arrivals (no bulk arrivals)
- An arrival to the queue (birth) causes the state to change by +1. A departure (death) causes the state to change by -1
- Below: State transition diagram with $n$ states, arrival rates $\lambda_n$ and service rates $\mu_n$. Arrival times and service times are exponentially distributed

$$
\begin{align*}
\lambda_0 & \quad \lambda_1 \\
\mu_1 & \quad \mu_2 \\
\lambda_2 & \quad \lambda_{j-2} \\
\mu_{j-2} & \quad \mu_{j-1} \\
\ldots & \quad \ldots \\
\lambda_{j-1} & \quad \lambda_j \\
\mu_{j-1} & \quad \mu_j \\
\lambda_j & \quad \lambda_{j+1} \\
\mu_j & \quad \mu_{j+1} \\
\ldots & \quad \ldots
\end{align*}
$$

Birth-Death Process

- The **steady-state probability** $p_n$ of a birth-death process being in state $n$ is given by the following theorem:

$$
p_n = p_0 \frac{\lambda_0 \lambda_1 \cdots \lambda_{n-1}}{\mu_1 \mu_2 \cdots \mu_n}
$$

$$
= p_0 \prod_{j=0}^{n-1} \frac{\lambda_j}{\mu_{j+1}}, \quad n = 1, 2, K, \infty
$$

- $p_0$ is the probability of being in the **zero state**
- Can be proven (see book)
- Now that we have an expression for state probabilities we are able to analyze queues in the form of M/M/m/B/K
- Based on the state probabilities we can compute many other performance measures
M|M|1 Queuing Model

- **Most commonly used** type of queue
- Can be used to model single-processor system or individual devices in a computer system
- Interarrival and service times are exponentially distributed, one server
- No buffer or population size limits, FCFS service discipline
- Analysis: We need the **mean arrival rate** \( \lambda \) and the **mean service rate** \( \mu \)
- State transition similar to birth-death process with \( \lambda_n = \lambda \) and \( \mu_n = \mu \)
- The probability of \( n \) jobs in the system becomes:

\[
p_n = \left( \frac{\lambda}{\mu} \right)^n p_0, \quad n = 1, 2, K, \infty
\]

The quantity \( \lambda/\mu \) is called **traffic intensity**
- It is usually denoted by \( \rho \). Thus \( p_n = \rho^n p_0 \)
- All probabilities should add to 1. Knowing this we can derive an equation for the probability of zero jobs \( (p_0) \) in the systems:

\[
p_0 = \frac{1}{1 + \rho + \rho^2 + L + \rho^\infty} = 1 - \rho
\]

- Substituting \( p_0 \) in \( p_n \) leads to:

\[
p_n = (1 - \rho)\rho^n, \quad n = 0, 1, 2, K, \infty
\]

- Based on this expression, many other properties can be derived
- Utilization of the server: \( U = 1 - \rho_0 = \rho \)
- The mean number of jobs in the system:

\[
E[n] = \sum_{n=1}^{\infty} np_n = \sum_{n=1}^{\infty} n(1 - \rho)\rho^n = \frac{\rho}{1 - \rho}
\]
The probability of \( n \) or more jobs in the system is:

\[
P(\geq n \text{ jobs in the system}) = \sum_{j=n}^{\infty} p_j = \sum_{j=n}^{\infty} (1 - \rho)^j = \rho^n
\]

Using Little’s law we can compute the mean response time:

\[
E[n] = \lambda E[r]
\]

\[
E[r] = \frac{E[n]}{\lambda} = \left(\frac{\rho}{1 - \rho}\right) \frac{1}{\lambda} = \frac{1}{\mu} \frac{1}{1 - \rho}
\]