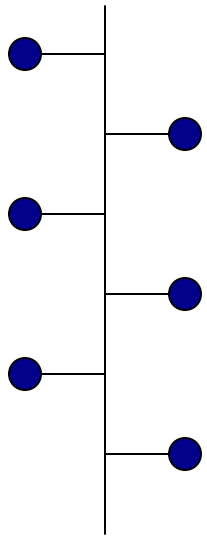
Unit 6
Queueing
Error Control



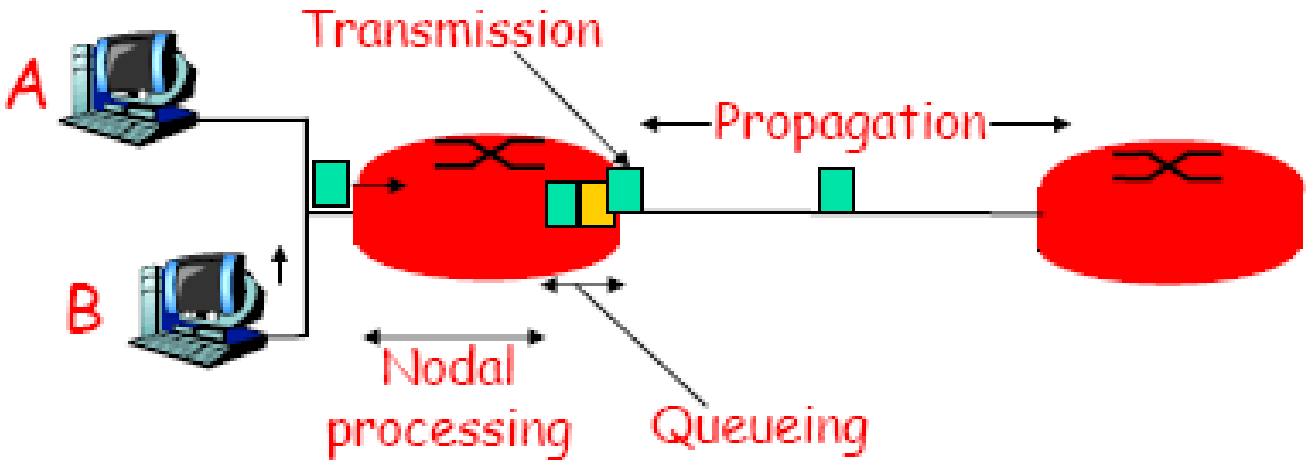
Acknowledgements – slides comming from:

- Slides from **Shivkumar Kalyanaraman (RPI) – gratefully acknowledged**
- Some figures have been used form the earlier issues of the EECS 122 tought by Prof Jean Walrand.
- **Data and Computer Communication** by Wiliam Stallings (our supplementary textbook) – numerous slides!
- **Data Communications and Networking** by B. **Forouzan**, Mc Graw Hill, 2004

Delay on the way – summary

[Source: Dr. Cheng Lehigh Univ.]

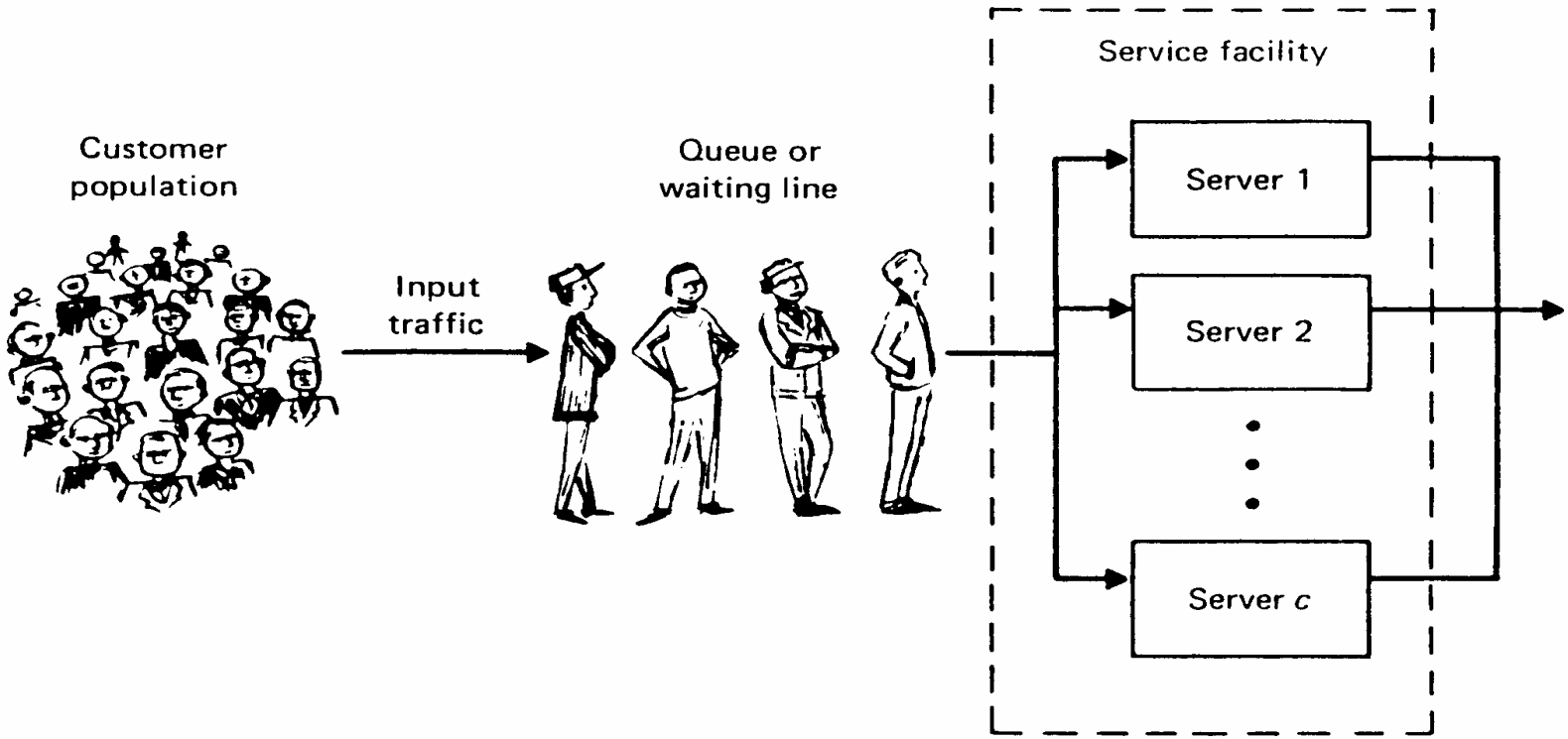
1. Nodal processing:
   - Check bit errors
   - Determine output
2. Queueing
   - Time waiting at output for trans.
   - Depends on congestion at router
3. Transmission delay:
   - R=link bandwidth (bps)
   - L=packet length (bits)
   - Time to send bits into link: L/R
4. Propagation delay:
   - d = length of physical link
   - s = propagation speed in medium
   - Propagation delay = d/s



Just to remaind you the issue of queueing...

So what is queueing about?

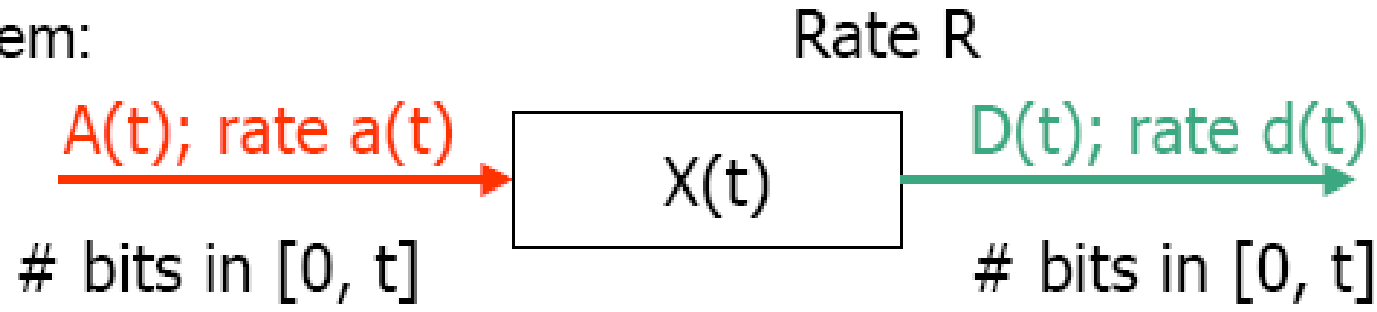
[Fig from O ALlen]

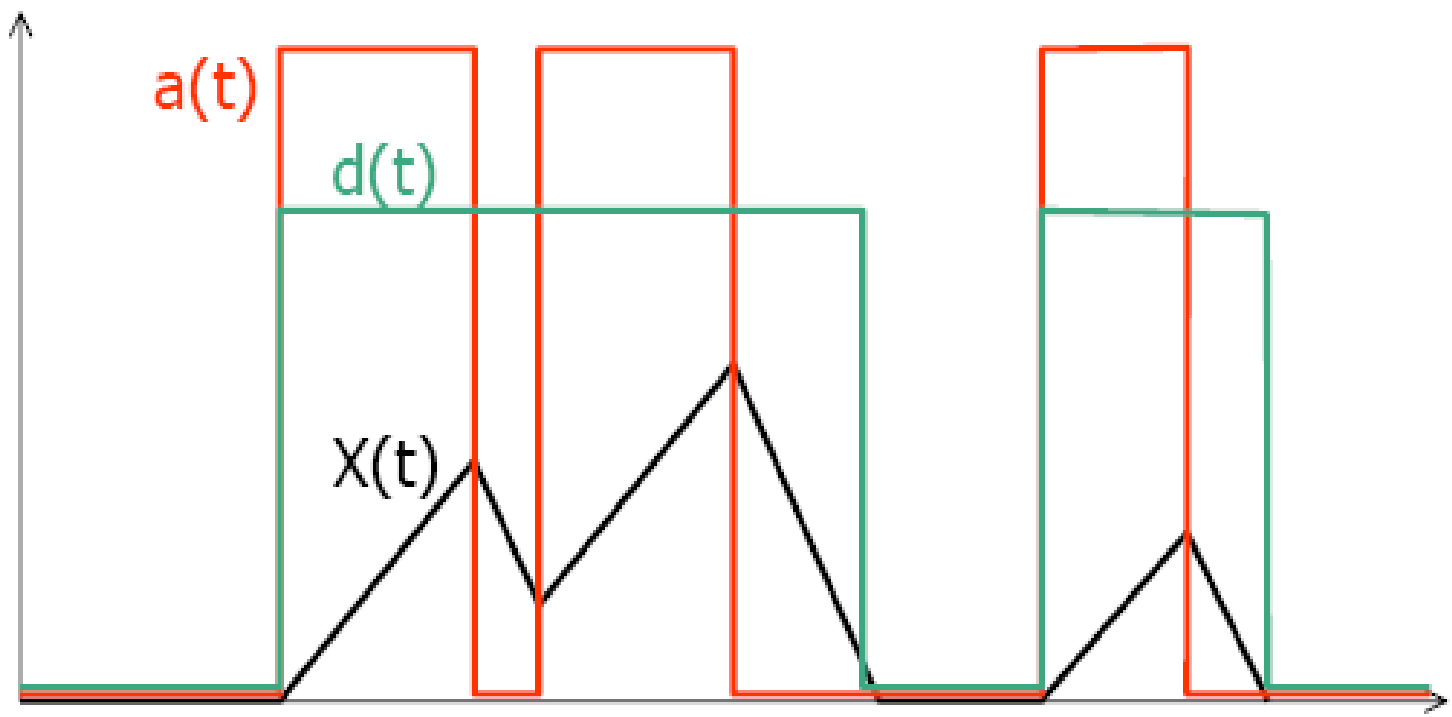


Whenever there is irregular demand for some „service“ taking a random time – there unavoidably appear queues....
Only fully deterministic demand, and fully deterministic service (which never happens!) would eliminate the queueing effect...

The Fluid view on the „amount of work“ [EECS122, Walrand]

- System:





Observe the change of the „work backlog“

Queueing systems parameters:

- Customers (eg. – packets!; phone calls!) arrive individually in discrete, randomly distributed time intervals.
- The service can be provided on a number of parallel (for simplicity identical) servers (e.g. Transmission lines)
- Each customer has a randomly distributed service time (e.g. Packet transmission duration, resulting out of packet length; e.g duration of a phone call).
- There might be a possibility to form a queue (e.g. Buffering of packets!) – but in some cases the customers experiencing all the servers busy might be just lost (busy line in telephone systems).

Kendall Notation for Queuing Systems

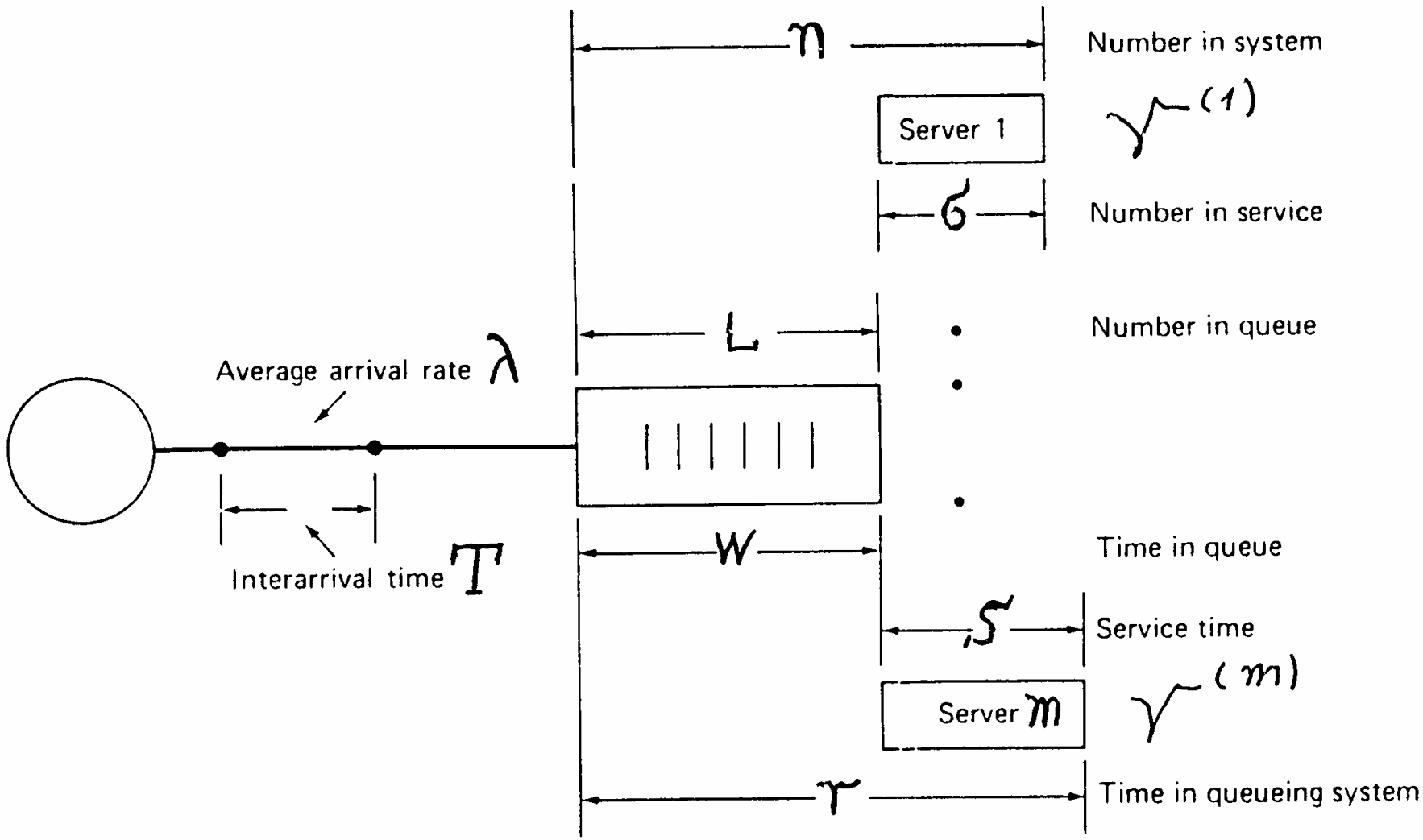
- Kendall Notation A/B/X/Y/Z
  - A is the interarrival-time distribution, B the service pattern
  - X the number of parallel service channels, Y the restriction on system capacity
  - Z the queue discipline

Characteristics	Symbol	Explanation
Interarrival-time distribution (A)	M	Exponential
	D	Deterministic
	E_k	Erlang type *k* (*k = 1, 2, ...*)
	H_k	Hyperexponential type *k*
	PH	Phase type
	G	General
Service-time distribution (B)	M	Exponential
	D	Deterministic
	E_k	Erlang type *k* (*k = 1, 2, ...*)
	H_k	Hyperexponential type *k*
	PH	Phase type
	G	General
Number of parallel servers (X)	1,2, ...	
Restriction on system capacity (Y)	1,2, ...	
Queue discipline (Z)	FCFS	First come, first serve
	LCFS	Last come, first serve
	RSS	Random selection for service
	PR	Priority
	GD	General discipline
	PS	Processor sharing
	RR	Round Robin

So , what would we like to know?

- There are values interesting for the customer:
  - The waiting time until the service begins
  - The complete time spend in the system (arrival to departure)
  - The probability of not obtaining the service ….
- There are values interesting to the service provider:
  - The utilization of the servers
    Comment: setting up the servers is usually an investment – each service brings some revenue – thus utilization is crucial…
  - The length of the queue (how should the buffers be dimensioned?)
- Basic challenge of the queuing theory: How to determine the distributions of the interesting values from the queuing system parameters…

Some important values



Some random variables used in queueing theory models.

Some useful abstractions

- An infinite amount of potential customers
  - Abstraction for a LARGE number of potential customers, or – in other words – for the case in which the length of the queue does not influence the arrival of new customers….
- An infinite queue length
  - Abstraction for the case of a “waiting capacity” being long enough
- The selected distributions of random variables
  - This is - for analytical studies – always a tradeoff between real, observed features and mathematical abstractions which make it possible to compute the interesting values.

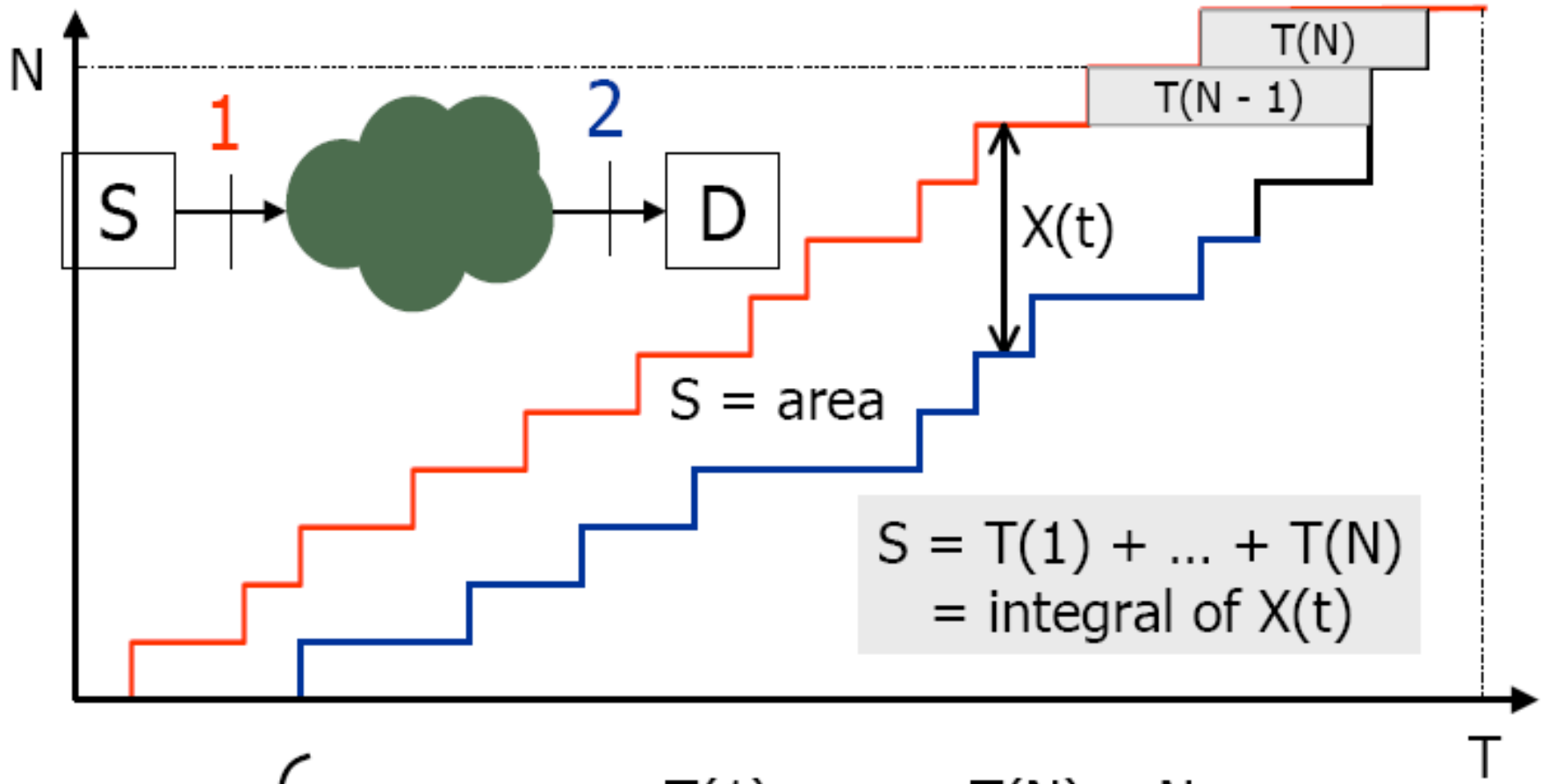
Unlimited Buffer Case - Some Formula

- The traffic intensity ρ $$\rho = \frac{\lambda E[S]}{m}$$
- A system with unlimited buffer size DOES NOT NECESSARILY have to be stable, i.e. It could happen that the queue „tends to grow indefinitely“
- When are such systems stable? If $\rho < 1$
- In this case server utilization $U_k = \frac{\lambda E[S]}{m}$ and throughput $\Lambda = \lambda$.

  Comment: If the system is stable the throughput Λ must equal the long-run rate at which customers arrive.
- Further in this considerations we will discuss only features of stable queuing systems

The Little´s Theorem

[EE122, Walrand]

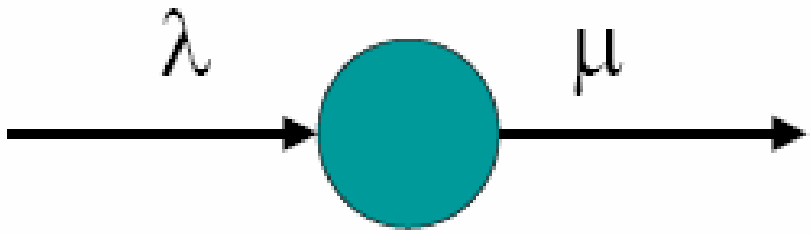


$$\frac{1}{T}\int X(t)dt = \frac{S}{T} = \frac{T(1) + \ldots + T(N)}{N} \cdot \frac{N}{T}$$

→ Average occupancy = (average delay)x(average arrival rate)

The basic queuing system

- Example: M/M/1 Queue



Arrivals are Poisson with rate λ

Service times are exponentially distributed with mean $1/\mu$

Average delay per packet

$T = 1/(\mu - \lambda) = (1/\mu)/(1 - \rho)$ where $\rho = \lambda/\mu$ = utilization

For instance, $1/\mu$ = 1ms and ρ = 80% => Q = 5ms

Some observations for the behavior...

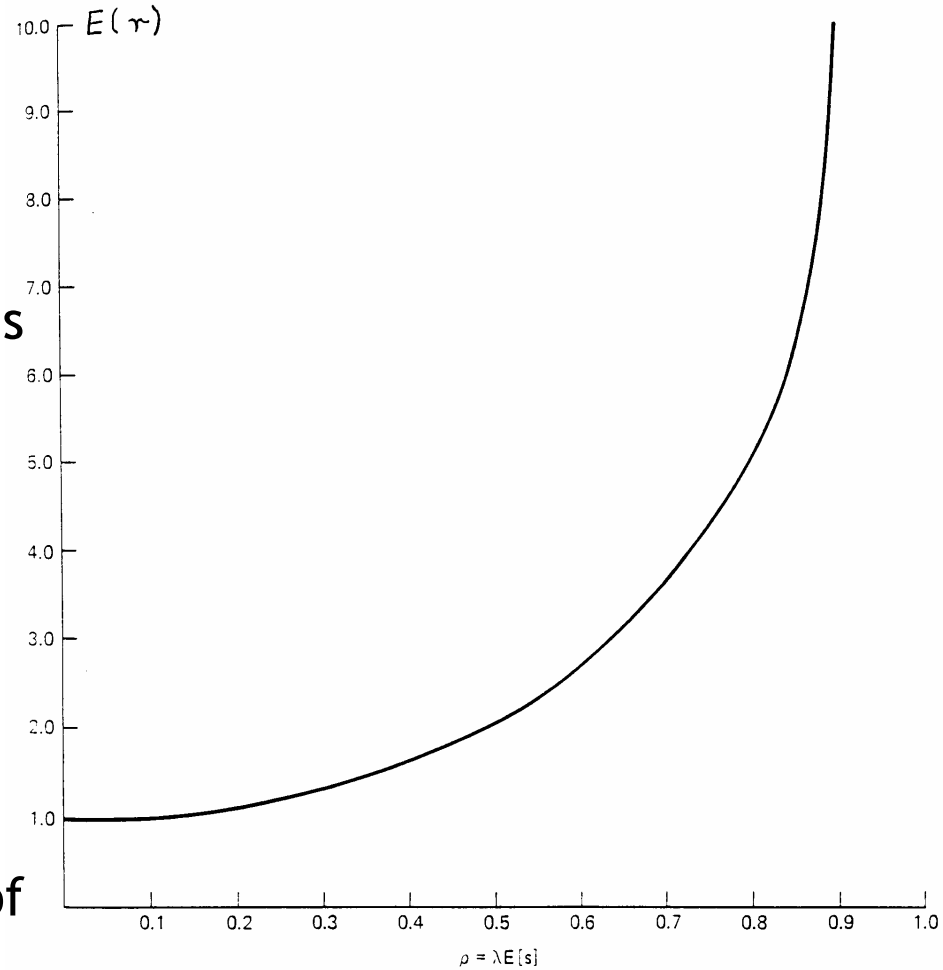
- That is, the slope of the curve increases rapidly as ρ grows beyond about 0.8. Since

$$\frac{dE[r]}{d\rho} = E[S](1-\rho)^{-2} = E[S](1-E[S]\cdot\lambda)^{-2}$$

a small change in ρ (due to a small change in λ, assuming *E[S]* is fixed) causes a change in *E[r]* given approximately by

$$\left(\frac{dE[r]}{d\rho}\right)\Delta\rho = \left(\frac{dE[r]}{d\rho}\right)E[S]\Delta\lambda = E[S]^2(1-\lambda E[S])^{-2}\Delta\lambda$$

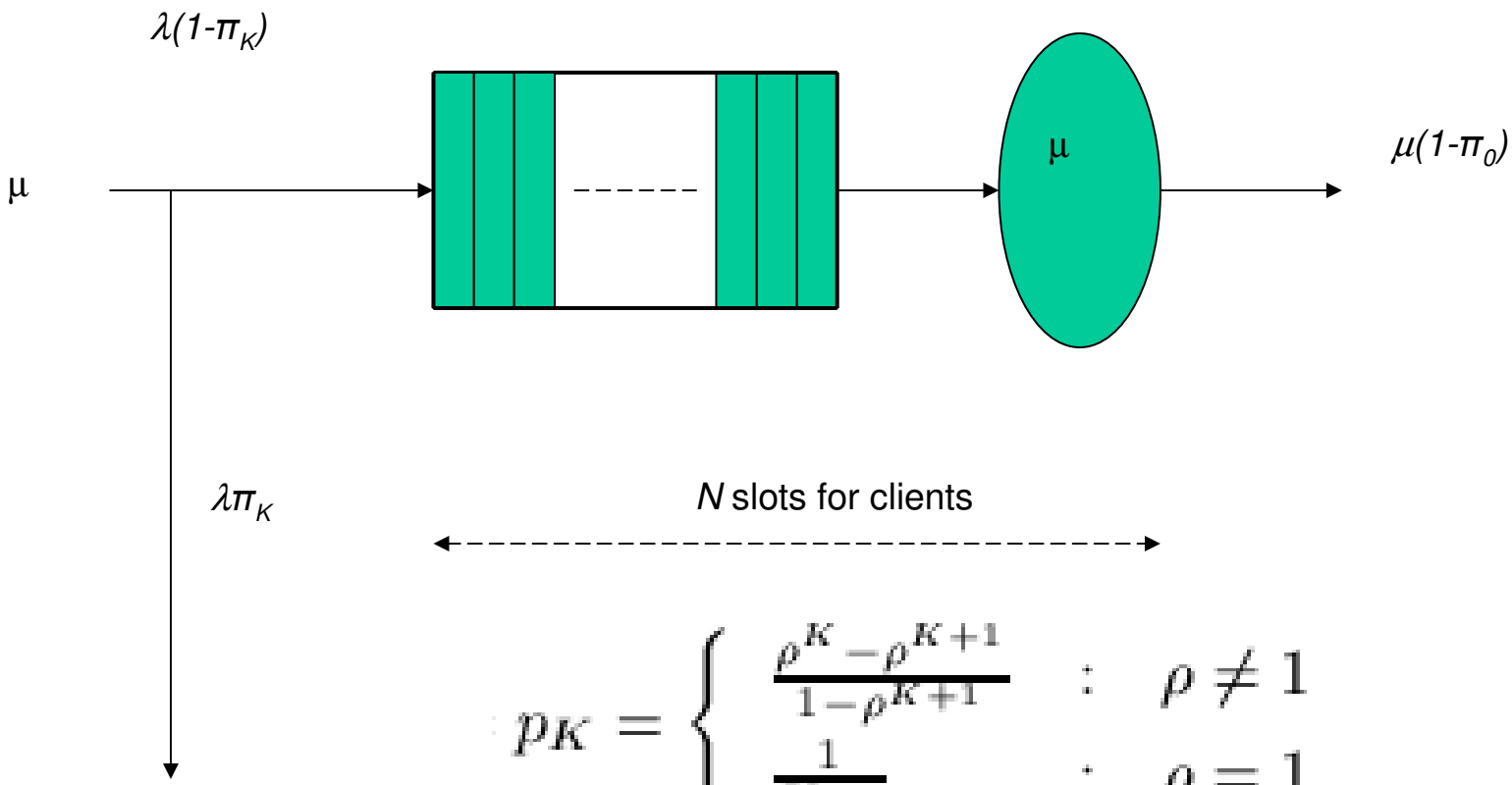
Thus, if $\rho = 0.5$, a change in $\Delta\lambda$ in λ will cause a change in *E[r]* of about $4E[S]^2\Delta\lambda$, while, if $\rho = 0.9$, the change in *E[r]* will be about $100E[S]^2\Delta\lambda$, or 25 times the size of the change that occurred for $\rho = 0.5$!



Normalized average time in the system, *E[r]/E[s]*, for M/M/1 queuing system.

M/M/1/K

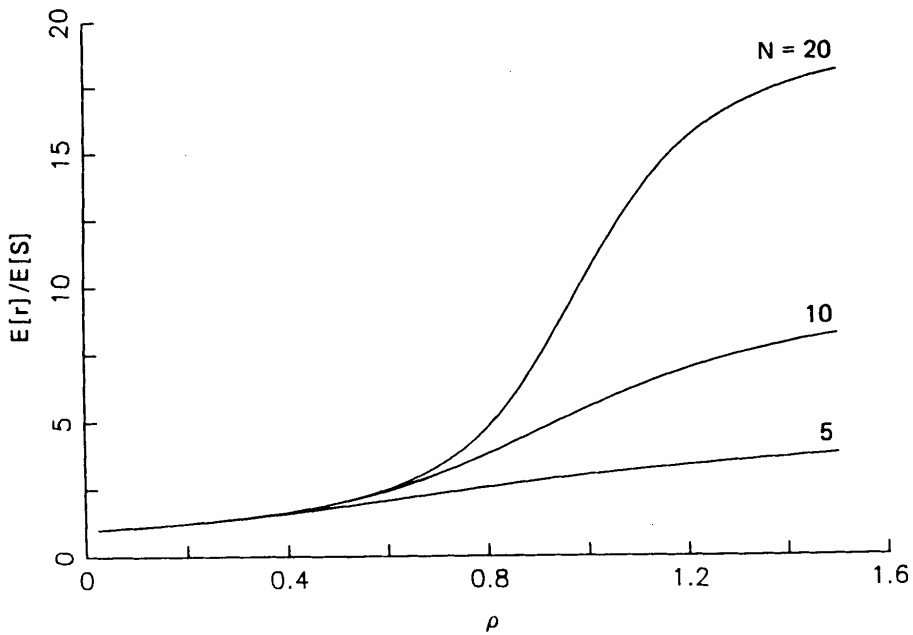
- What happens if we take a more realistic case of limited buffer capacity: all in all only N customers are allowed in the M/M/1/K queuing system
- .

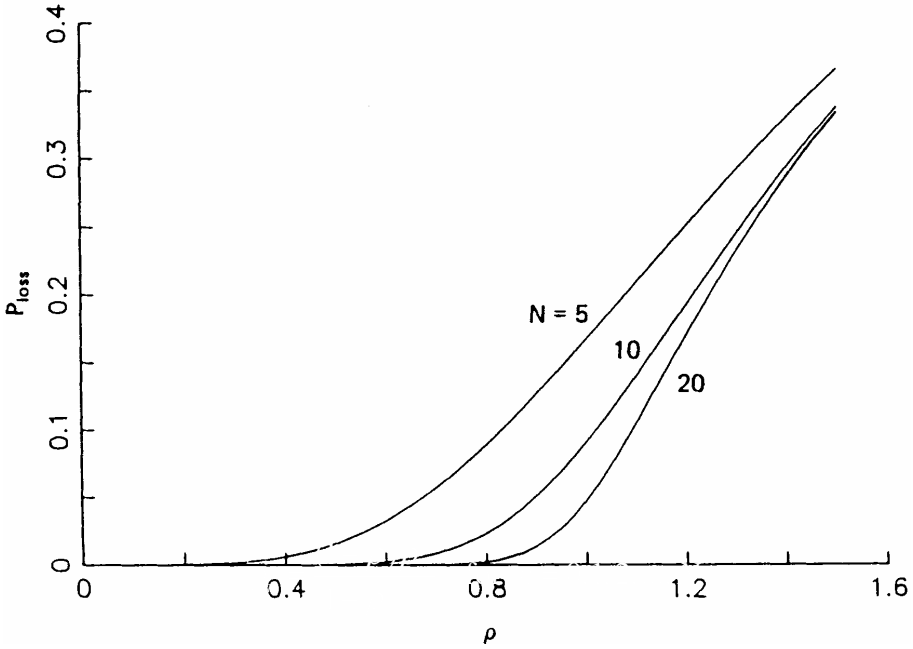


$$p_K = \begin{cases} \frac{\rho^K - \rho^{K+1}}{1-\rho^{K+1}} & : \quad \rho \neq 1 \\ \frac{1}{K+1} & : \quad \rho = 1 \end{cases}$$

Probability of customer loss

M/M/1/K





- Due to the losses- this system is ALWAYS stable, independent of the arrival rate/serivce rate ratio!!!
- The normalized mean delay tends asymptotically to N =K
- Be aware: considering the normalized mean delay independently from the loss probability curves is not resonable
- One typical question: How big should the bufer be, to assure loss probabiliy smaller than some treshold....

The case of multiple servers...

Comparison of Single Server and Multiserver Queues

a/ a service center having *a single server*

$$E[r(1, \lambda, \gamma)]$$

b/ a service center having *m servers*, each of which is *m times slower* than the server in a/

$$E[r(m, \lambda, \gamma/m)]$$

c/ a service center having *a single server* which is *m times slower* than the server in a/ and where the service center handles *(1/m)th* of the customers.

$$E[r(1, \lambda/m, \gamma/m)],$$

The following inequalities hold:

$$E[r(1, \lambda, \gamma)] < E[r(m, \lambda, \gamma/m)] < E[r(1, \lambda/m, \gamma/m)],$$

The inequalities are valid also for the M/G/m case, although the first one only for variability coefficients smaller than one.

Comparison of three systems – an example

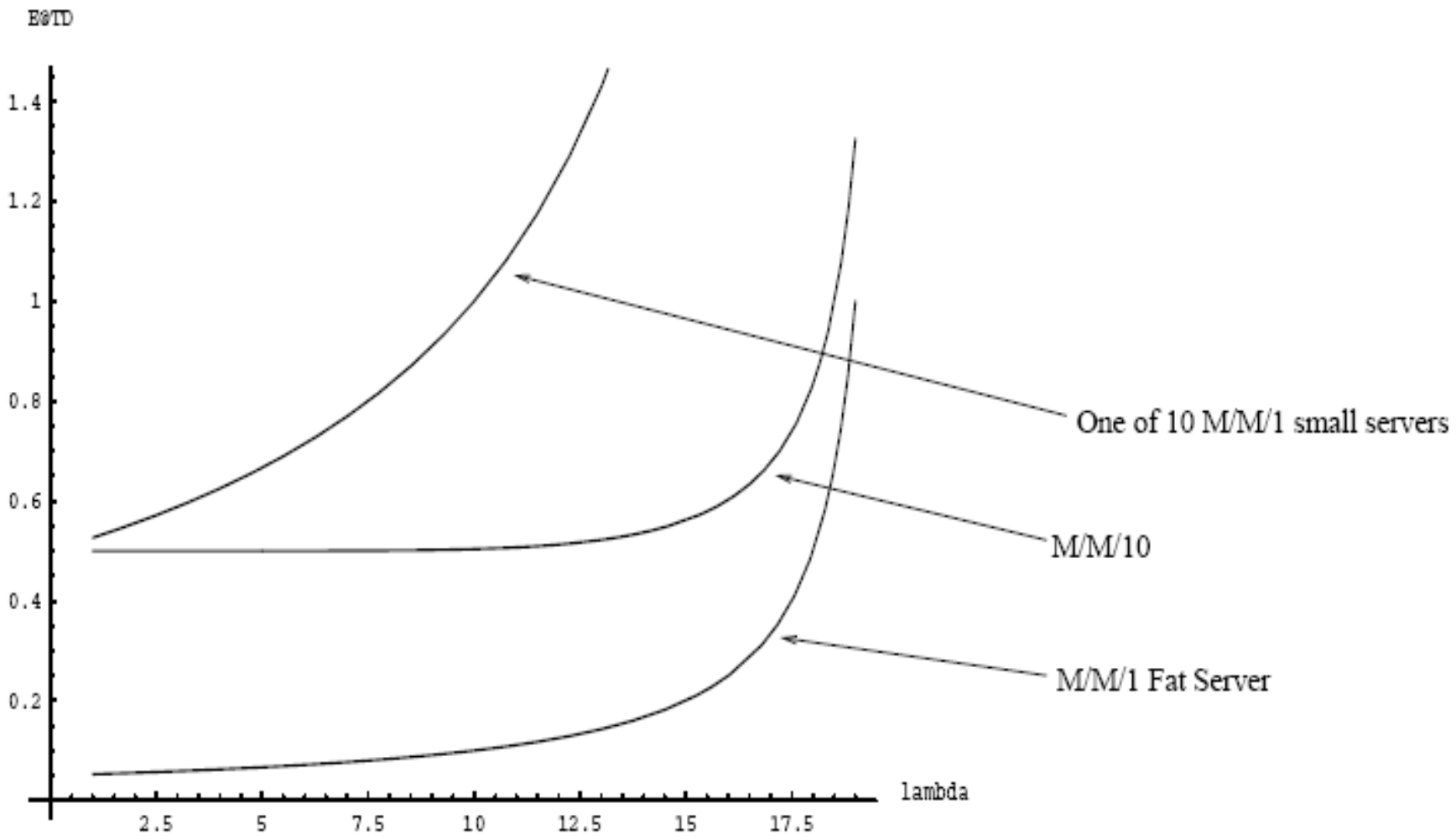
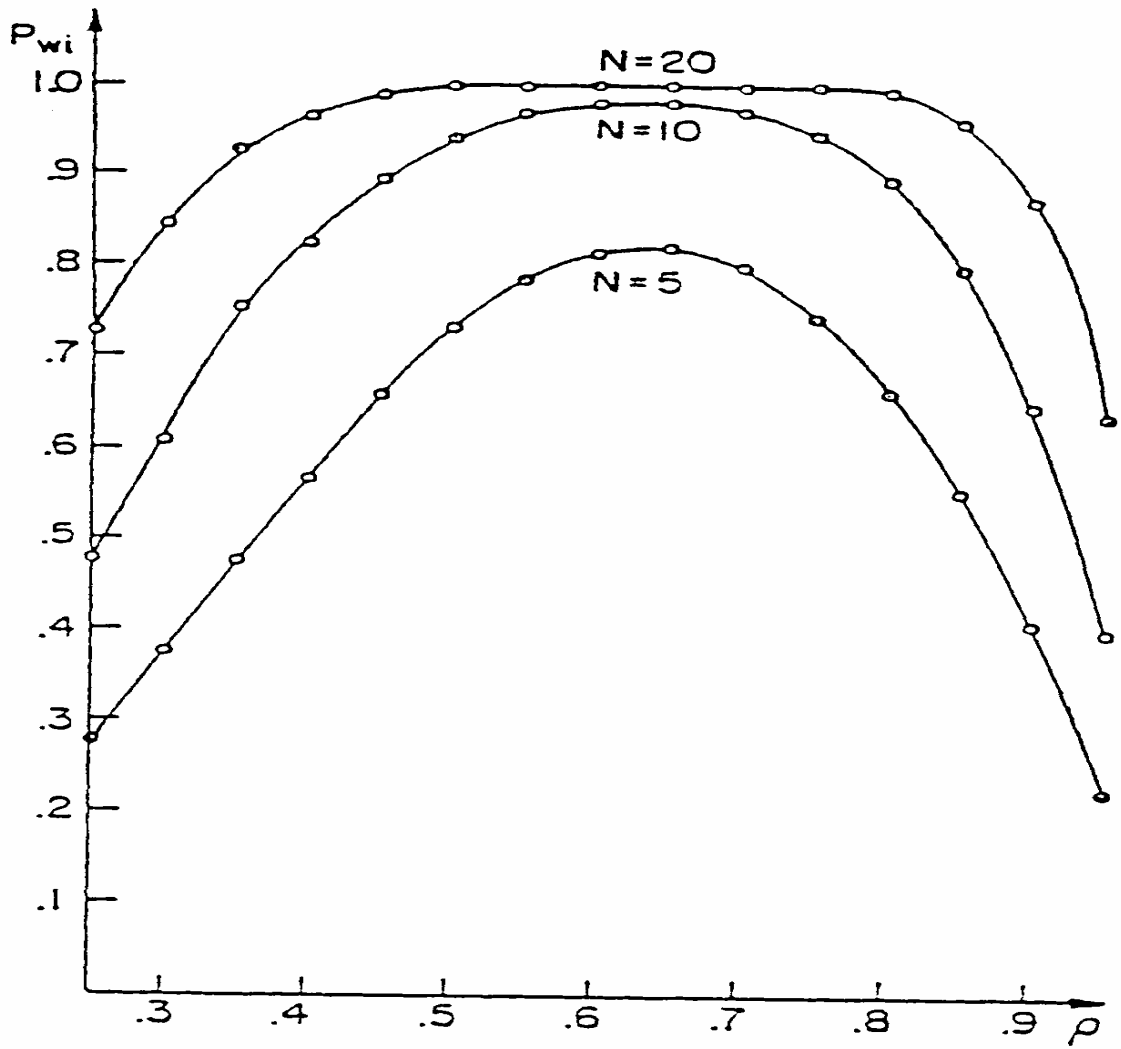


Figure 2.10: Mean Response Times for three different systems

Some explanations...the case of N separate queues

- It is better to use a system with single queue and multiple servers than support a separate queue for each server..

In the plot:
The probability of Having at least one Idle server in spite of at leat one non-empty queue in the system of N servers with splitted queues



Back to the Telephone System... Traffic Characterization

- **Call arrivals**
  - **a random process**
- **Call holding time**
  - **a random variable**
  - **duration of occupancy of a traffic path by a call**
- **Traffic intensity not the same throughout the entire day**
  - **design decisions usually based on traffic intensity during the busy hour (worst case)**
- **Busy hour is the 1-h period of the day during which traffic intensity is highest**
  - **CCITT : "The busy hour refers to the traffic volume or number of call attempts, and is that continuous 1-h period lying wholly in the time interval concerned for which this quantity is greatest."**

Measures of telephony traffic

- **Telephone traffic is the aggregate of telephone calls over a group of circuits or trunks with regard of the number and duration of calls.**
- **We measure traffic intensity (A) as : $A = \lambda t_m$, where**
  - **λ is the average arrival rate (ex., calls/hr)**
  - **t_m is the average holding time (ex., hrs)**
- **Units:**
  - **Erlangs (dimensionless) : calls-second per second**
  - **CCS : hundred (century) calls-second per hour**
  - **1 erlang = (60)(60)/100 = 36 CCS**
- **Capacity of a single channel is one erlang**
  - **Interpretation: a telephone that is busy 10% of the time represents a load of 0.1 erlang on that particular line**
- **Example: 2 calls/hour with average holding time of 5 minutes**
  - **what is the traffic intensity in Erlangs? In CCS?**

Blocking and handling of lost calls

- **Grade of service (an estimate of blockage probability):**

$$p = \frac{Number\ of\ lost\ calls}{Number\ of\ offered\ calls}$$

- **How are lost calls handled?**
  - **Lost calls held : user will immediately reattempt the call on receipt of a congestion signal**
  - **Lost calls cleared : user will hang up and wait some time interval before reattempting the call**
  - **Lost calls delayed : user is put in a queue, and waiting calls are handled FIFO (or LIFO, or randomly)**

Erlang_B Formula.

Assumptions:

- Arrivals from an infinite Poisson source (the inter-arrival times are exponentially distributed).

This corresponds to a situation when number of customers is much larger than the number of resources available to service them. Acceptable results if the number of customers is at least 10 times the total number of resources (N).

- Calls which cannot be served are lost (and do NOT appear again)

- **Probability of blockage at the switch due to congestion or "all trunks busy":**

$$E_B = \frac{\frac{A^n}{n!}}{\sum_{x=0}^{n} \frac{A^x}{x!}}$$

- **A is the mean of the offered traffic [Erlangs], n is the number of trunks**

Addendum: Denotations.

- T — interarrival time (a random variable)
- $\lambda = 1/E[T]$ — arrival rate
- m — number of parallel server
- S — service demand / time per customer without regard to customer type (a random variable)
- $\mu = 1/E[S]$ — service rate
- n — steady state number of customers in the service center (a random variable)
- r — steady state response time (a random variable)
- U — utilization of server
- Λ — throughput of customers

Addendum:Random Variables – denotations

- For a continuous random variable X
  - $\overline{X}, E[X]$ mean
  - $Var[X]$ variance
  - $C[X]$ coefficient of variation $$C[X] = \frac{\sqrt{Var[X]}}{E[X]}$$
  - $F_x(t)$ probability distribution function
  - $f_x(t)$ probability density function

Bit Error Detection /Correction

Error Hypothesis (1)

- The usual assumption for the analysis:
  Independent single bit errors - each bit is in error with given, equal probability p. This means that for an *n* bit long frame the probability of error-free transmission is $(1-p)^n$. On the other hand, the probability of at least one bit being in error within the frame is $P_{fr} = 1 - (1-p)^n$, which for $p \ll 1$ tends to $n*p$.

- Typical bit error rates (BER) of transmission lines used for telecommunication:
  - X-25 Networks BER = 10^{-5}
  - Local Area Networks BER = 10^{-7}
  - Modern Fiber Optics BER = 10^{-9}

Error Hypothesis (2)

- In reality:
  Errors tend to appear in bursts: this follows from the physical characteristics or reasons, like electrical disturbances, loss of bit synchronization, cross-talks among transmission channels, etc. bursty errors are harder to model and to recognize. Less frames are corrupted if errors occur in bursts.

- Examples:
  - Within data transmission over telephone lines, within each 8 bits there are at least as many double errors as single errors.
  - In a 1.544 Mb/s digital link (T1) during one month, errors have been observed only in 8 days, all grouped in 19 bursts, each shorter than 50ms.

Error detection/correction- open loop

- What can be done using an open loop for a single frame (without feedback from the receiver to the transmitter) - the case of connectionless data transmission?

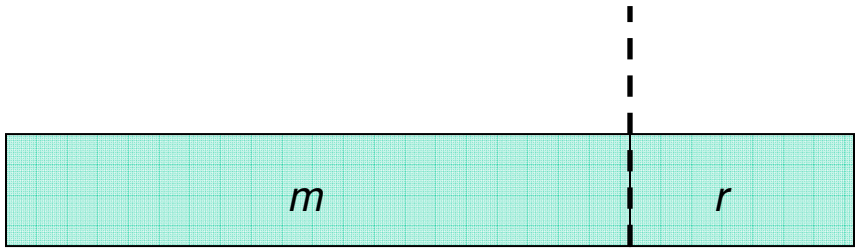
- Redundancy can be used within the frame in order to detect the error or even to correct the errors.
  Because of this redundancy a frame consists of more bits than the sum of the header (envelope) and user information.
  Significantly higher redundancy is needed for error correction than for error detection.

- Idea: a message of length m bits is extended by a redundancy string of r bits, forming a data packet of n=m+r bits

Block codes

– Basic concepts (block codes):

- m bits of data are to be transmitted in a transparent way, i.e. each of the 2^m possible binary sequences (code words) is permitted. Therefore, in case of error, a code word WR different from the transmitted code word WT is received. The receiver cannot recognize the error.
- the original string of "m" bits is transformed into a string of length "n", the additional n-m bits representing the redundancy.Only 2^m out of all 2^n possible code words are permitted. Arrival of a prohibited code word represents an error.



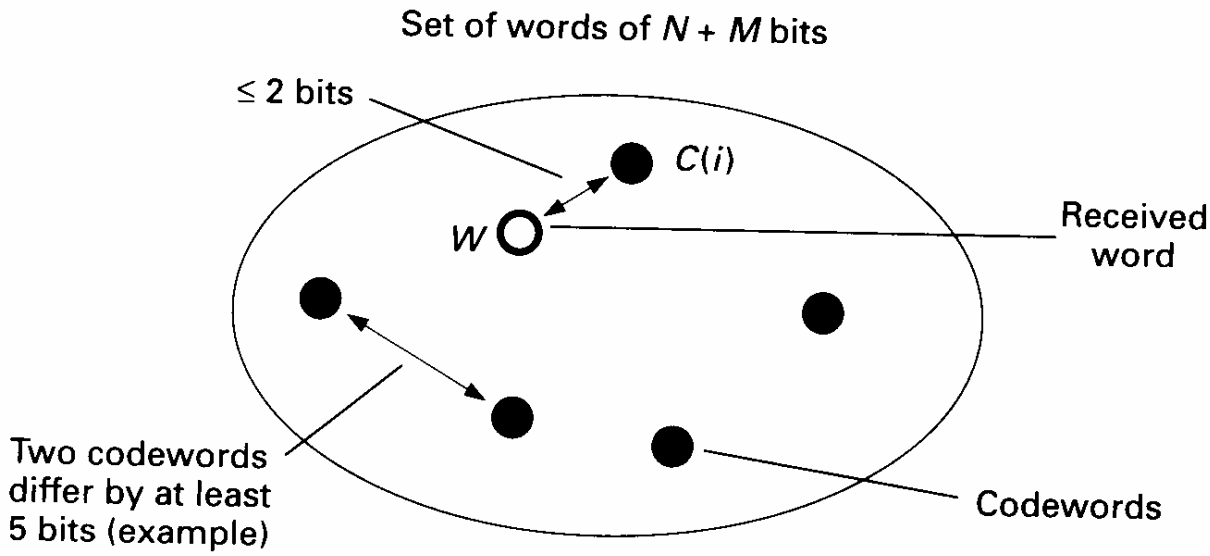
m bits to be sent (original message)

The group of *r* error-control bits is added

A simple parity: n=m+1, the number of "1"s in the sequence has to be either odd or even. Only odd number of bit errors can be detected.

The Hamming Distance

- Let i and j be two code words. The Hamming distance *d(i,j)* is the number of bits in which they differ. For any code *C* there is an important parameter: Minimal Hamming distance between any two code words $d_{min}(C) = min\ d(i,j)$ over all *i,j*.



The received codeword is in error, if it does not belong to the set of legal codewords. We could try to guess what it could have been...

Block codes: What can we do?

- Error detection power of block codes:

  A code *C* can detect any combination of *x* or fewer errors, if $d_{min}(C) \geq x + 1$.

  – Justification: It requires at least $x+1$ errors to transform a transmitted word into an other legitimate word.

  – Good News: Many error patterns corresponding to more than *x* errors can also result in a prohibited word!!!

- Error correction power of block codes

  A code C can correct any combination of *y* or fewer errors, if $d_{min}(C) \geq 2y + 1$.

  – After receiving the prohibited word W_p, the unique legitimate word with minimum Hamming distance from W_p is selected.

Block Codes: What can we do? (2)

- Combined error detection/error correction power

  A code *C* can detect any combination of *x* or fewer errors and correct any combination of y or fewer errors if

  $$d_{min}(C) > x + y + 1.$$

- Remarks:
  - A high overhead is needed for error correction.
  - There is always a positive probability of an undetected/uncorrected error!

An Example: Hamming *(7,4)*

- An example: features of a Hamming *(7,4)* code
  - The minimum distance of three can be achieved by introducing $n-m=3$ redundancy bits.
  - Let $m1$, $m2$, $m3$, $m4$ represent the bits of the message, while $f1$, $f2$, $f3$ represent the redundant bits, chosen so as to satisfy

$$f1@m1@m3@m4 = 0$$
$$f2@m1@m2@m3 = 0$$
$$f3@m2@m3@m4 = 0$$

  - where @ stands for a modulo 2 addition (exclusive OR), identical to an modulo 2 subtraction.

Error detection/correction – some numbers

Error Detection Performance of Hamming (7,4) Code Assuming Independent Errors on Communications Link.

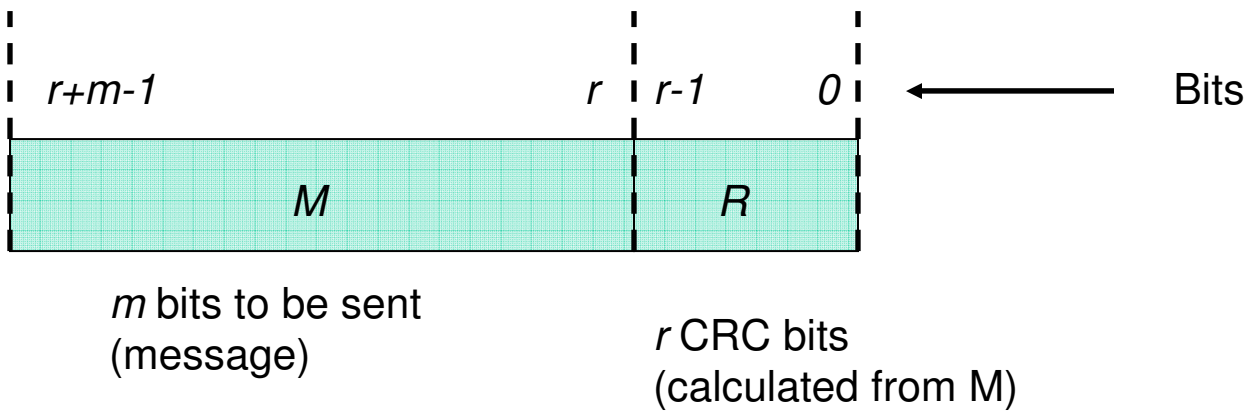
Bit Error Rate	Probability of Error in Transmission	Probability of Detection Failure Given Error	Probability of Undetected Error
10^{-5}	7.0×10^{-5}	1.0×10^{-10}	7.0×10^{-15}
10^{-4}	7.0×10^{-4}	1.0×10^{-8}	7.0×10^{-12}
10^{-3}	7.0×10^{-3}	1.0×10^{-6}	7.0×10^{-9}
10^{-2}	6.8×10^{-2}	1.0×10^{-4}	6.8×10^{-6}
10^{-1}	5.2×10^{-1}	1.0×10^{-2}	5.1×10^{-3}

Error Handling Performance of Hamming (7,4) Code When Used as Error Correcting Code, Assuming Independent Errors on Communication Link.

Bit Error Rate	Probability of Error in Transmission	Probability of Erroneous Decoding Given Error	Residual Error Rate
10^{-5}	7.0×10^{-5}	3.0×10^{-9}	2.1×10^{-13}
10^{-4}	7.0×10^{-4}	3.0×10^{-8}	2.1×10^{-12}
10^{-3}	7.0×10^{-3}	3.0×10^{-7}	2.1×10^{-11}
10^{-2}	6.8×10^{-2}	3.0×10^{-6}	2.0×10^{-8}
10^{-1}	5.2×10^{-1}	2.9×10^{-5}	1.5×10^{-6}

Efficient approach to error detection

- Question: How to achieve an error DETECTION with
  - Overhead independent of the codeword length
  - Low computational complexity at sender and receiver
  - High detection probability of multiple errors
- Answer: Polynomial codes



Message with CRC bits.
The group *R* of *r* error-control bits is added to the message *M* of *m* bits to be transmitted.

Polynomial Codes (1)

- Some polynomial $G_r(x)$ of the rank r, r being the number of redundancy bits r=n-k (called Generator polynomial) has to be agreed between the transmitter and receiver.
- The individual bits of the message to be sent (bit sequence of length k) are interpreted as coefficients of a polynomial $V_{k-1}(x)$ of rank (k-1).
  - e.g. 1,1,0,0,0,1 represents a polynomial $x^5 + x^4 + x^0$
- The expression $x^rV_{k-1}(x)$, corresponding to appending r zero bits to the original frame, is constructed.
- This expression is divided by G(x) using modulo 2 division. A quotient Q(x) and a remainder R(x) - a polynomial of rank not larger than (r-1) - is computed.
- A sum $T(x)= x^rV_{k-1}(x) + R(x)$ is built by substituting the last r positions of $x^rV_{k-1}(x)$ with the coefficients of the R(x). This is equivalent to subtracting R(x) modulo 2!!! This procedure is frequently called Cyclic Redundancy Check (CRC).

Polynomial Codes (2)

- Calculation of the Polynomial Code Checksum
  - Frame: 1101011011
  - Generator: 10011
  - Message: 11010110110000

```
                1100001010
10011 | 11010110110000
        10011
         10011
         10011
          00001
          00000
           00010
           00000
            00101
            00000
             01011
             00000
              10110
              10011
               01010
               00000
                10100
                10011
                 01110
                 00000
                  1110 ← Rest
```

Polynomial Codes (3)

- Attention
  - $x^r V_{k-1}(x) = Q(x)\ G(x)\ @\ R(x)$

    thus

    $T(x) = x^r V_{k-1}(x)\ @\ R(x) = Q(x)\ G(x)$
- The coefficients of $T(x)$, a multiple of $G(x)$ are transmitted as the codeword. The receiver has only to check if the polynomial $T^*(x)$, corresponding to the received bit sequence remains a multiple of $G(x)$, i.e. if the remainder of division by $G(x)$ is equal to 0.
- The key issue: choosing a good $G(x)$.
  - $G16\ ISO(x) = x^{16} + x^{15} + x^{12} + 1$
  - $G16\ CCITT(x) = x^{16} + x^{12} + x^5 + 1$
  - $G32(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

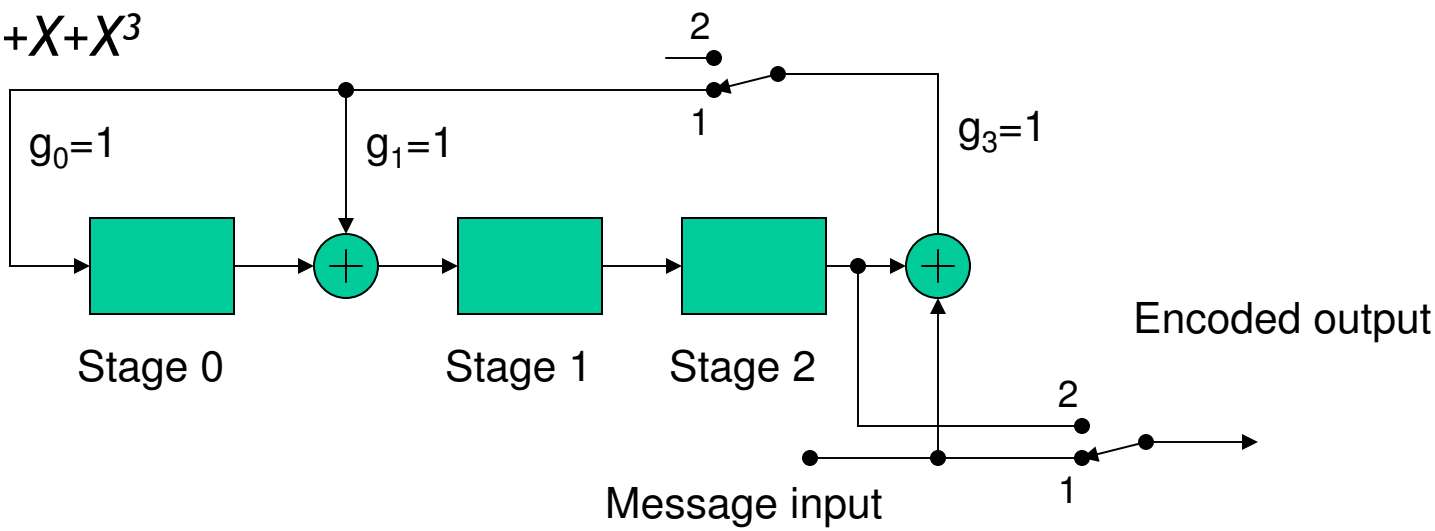
Polynomial Codes (4)

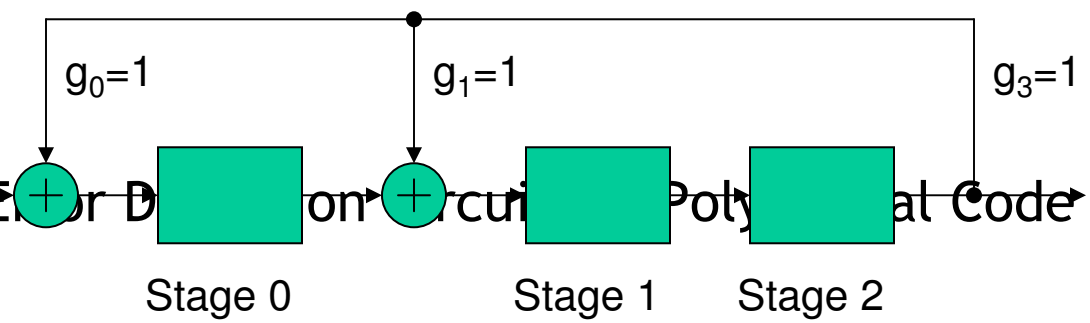
- Properties for a properly chosen *G(x)*
  - Detection of all:
    - single errors
    - pairs of isolated errors, if x does not divide $G(x)$ and $G(X)$ does not divide x^k+1 for any k up to the maximum frame length
    - any odd number of errors
      if $G(x)$ has $x + 1$ as a factor
    - all burst errors of length up to r
    - burst errors with length exactly equal to $r+1$:
      with probability $P = 1 - (1/2^{)(r-1)}$
    - burst errors of length greater than $(r+1)$
      with probability $P = 1 - (1/2)^r$

- Implementation issues
  - Can be implemented easily using shift registers with proper feedback!!
  - The implementation in hardware makes it possible to compute the checksum i.e. check the correctness on the fly, while transmitting/receiving on a serial link.
  - Implementations are available up to 140Mbits/s.
  - Identification of the position of CRC bits in the frame.

Implementation of the Checksum Calculation

- Shift Register Encoder for Polynomial Code with
- $G(X) = 1+X+X^3$





- Shift Register Encoder Decoder Circuit for Polynomial Code
- $G(X) = 1+X+X^3$

The Open Loop: Summary

- Error correction by use of redundancy in open loop is called **F**orward **E**rror **C**orrection: FEC)
- Features of FEC
  - No additional delay in data transmission
  - Significant increase in the volume of data
  - Not necessarily simple processing
- Used in satellite lines, high speed networks (?)

Look ahead: Closed Loop...ARQ

- Alternative approach: Automatic Repeat reQuest (ARQ)- a simple error detection and retransmission (at least of the information in error). There is a feedback from the receiver to the transmitter:
  - Some additional data exchange between the transmitter and receiver is needed in order to identify the frames in error.
  - A delay in information transfer is induced by the retransmission.
- ARQ scheme can be useful only assuming the following features:
  - A very small increase in data volume is sufficient for a low probability of undetected errors.
  - Some schemes of error-detection coding call for relatively small processing overhead.
  - Polynomial codes, Fletcher codes