Generating Random Variates I

Holger Füßler

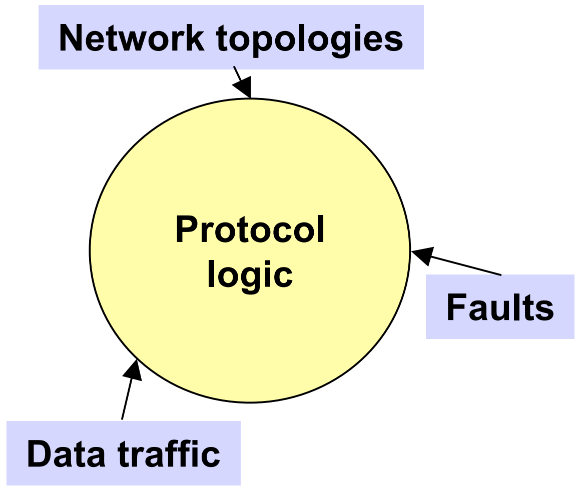
Course overview

1. Introduction	7. NS-2: Fixed networks
2. Building block: RNG	8. NS-2: Wireless networks
3. Building block: Generating random variates I and modeling examples	9. Output analysis: single system
4. Building block: Generating random variates II and modeling examples	10. Output analysis: comparing different configuration
5. Algorithmics: Management of events	11. Omnet++ / OPNET
6. NS-2: Introduction	12. Simulation lifecycle, summary

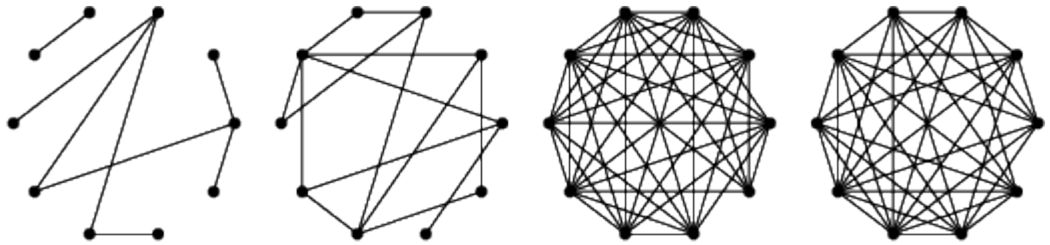
Structure of this lecture

- **Part I: Modeling examples w.r.t. U(0,1)**
  - Random graphs (topologies)
  - Constant bit rate data traffic with jitter
  - Randomness as protocol element
    - Example: multiaccess communication
- **Part II: Generation of other random variates for modeling network elements**
  - Inverse transform method
  - Example: exponential distribution
  - Inverse transform method for discrete random variates
  - Generalized inverse transform
- **Part III: Modeling examples w.r.t. exponential distribution**
  - Random direction mobility
  - On/off sources for generating bursty traffic

Random effects:
'how to' and 'what'



I Random graphs I



Eric W. Weisstein. "Random Graph." From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/RandomGraph.html

For each pair of vertices (u,v):

- » **There is an edge between u and v with probability a 2 [0,1].**
- » **Required: sampling from Bernoulli random variable; mass function:**

$$p(x) = \begin{cases} 1-a & \text{if} \quad x=0 \\ a & \text{if} \quad x=1 \\ 0 & \text{otherwise} \end{cases}$$

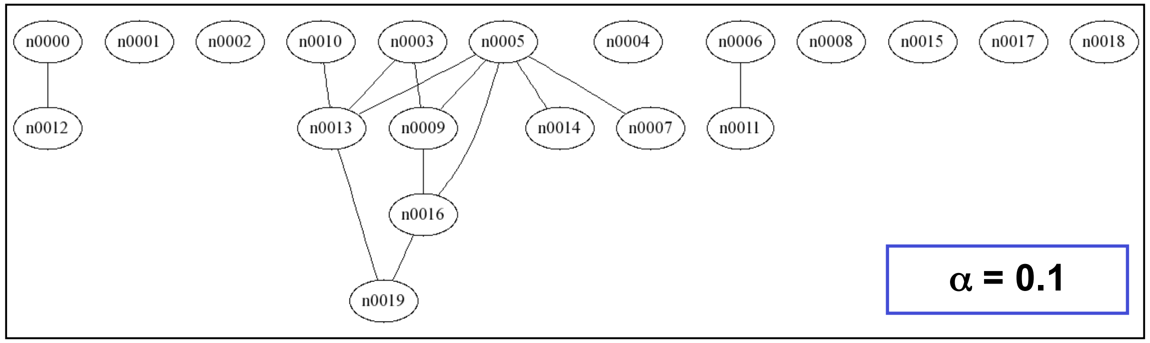
I Random graphs I – Example Program

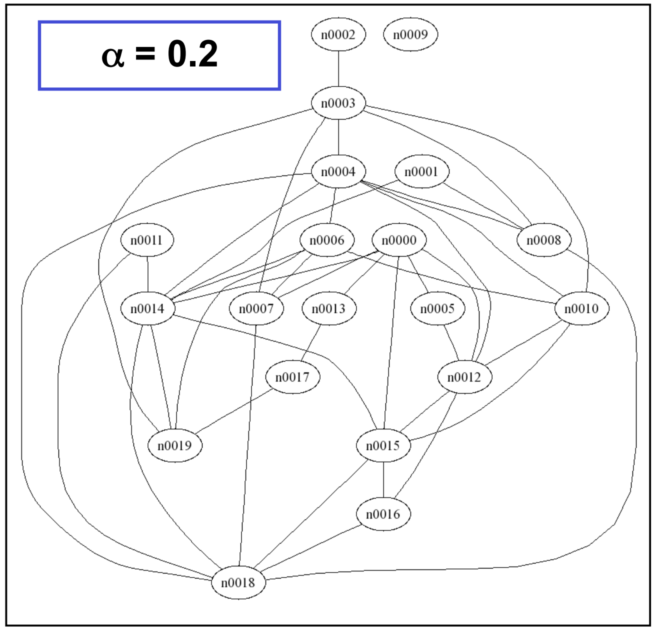
randgraph.c (in download area):

```
$ ./randgraph 0.2 > randgraph-02.dot

$ dot –Tps randgraph-02.dot -o randgraph-02.eps
```

Create Graph Description in dot format

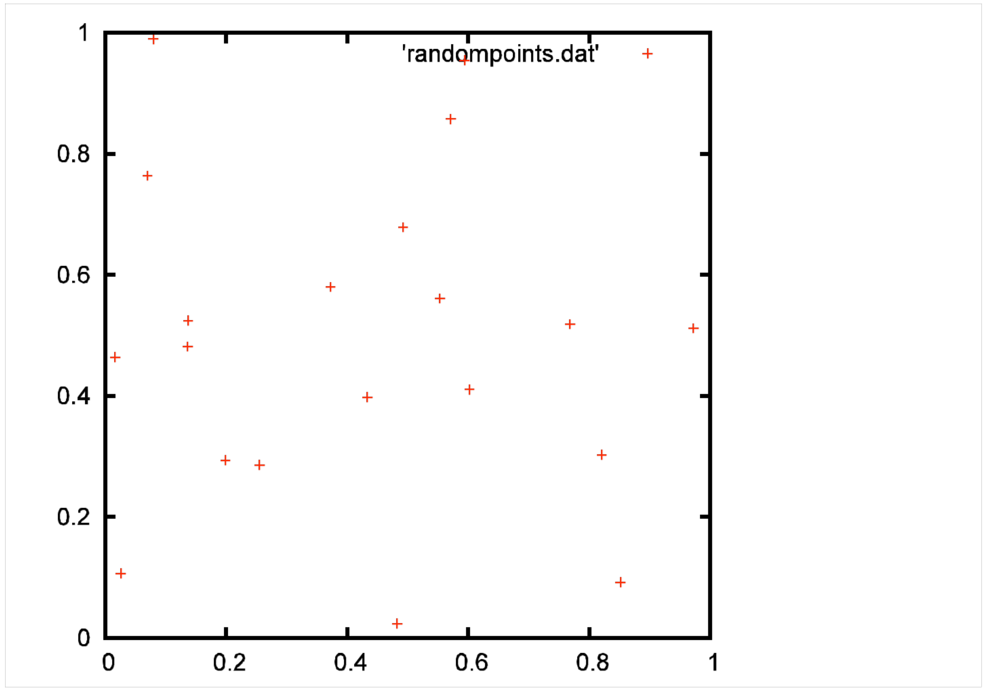
Create Picture (EPS)



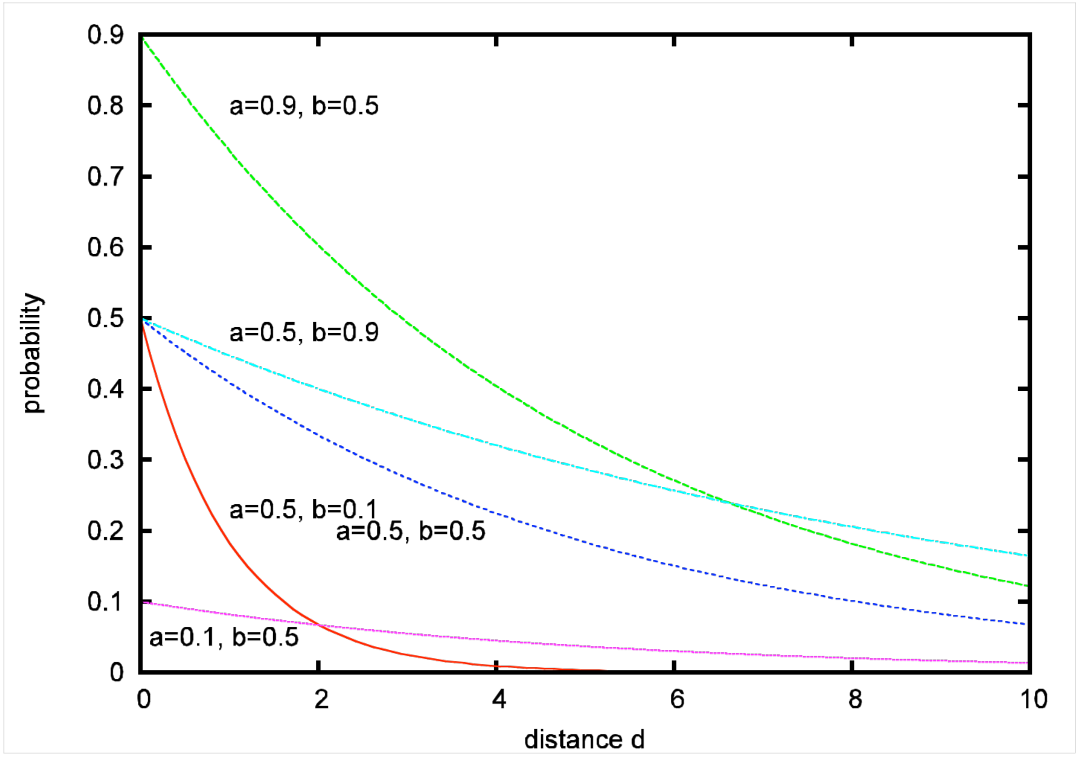


I Random graphs II

- **Select vertices by sampling from a 2-dimensional uniform distribution**
  - **Since dimensions should be independent, one can simply sample x and y coordinates from a 1-dimensional uniform distribution**



I Random graphs III: Waxman model

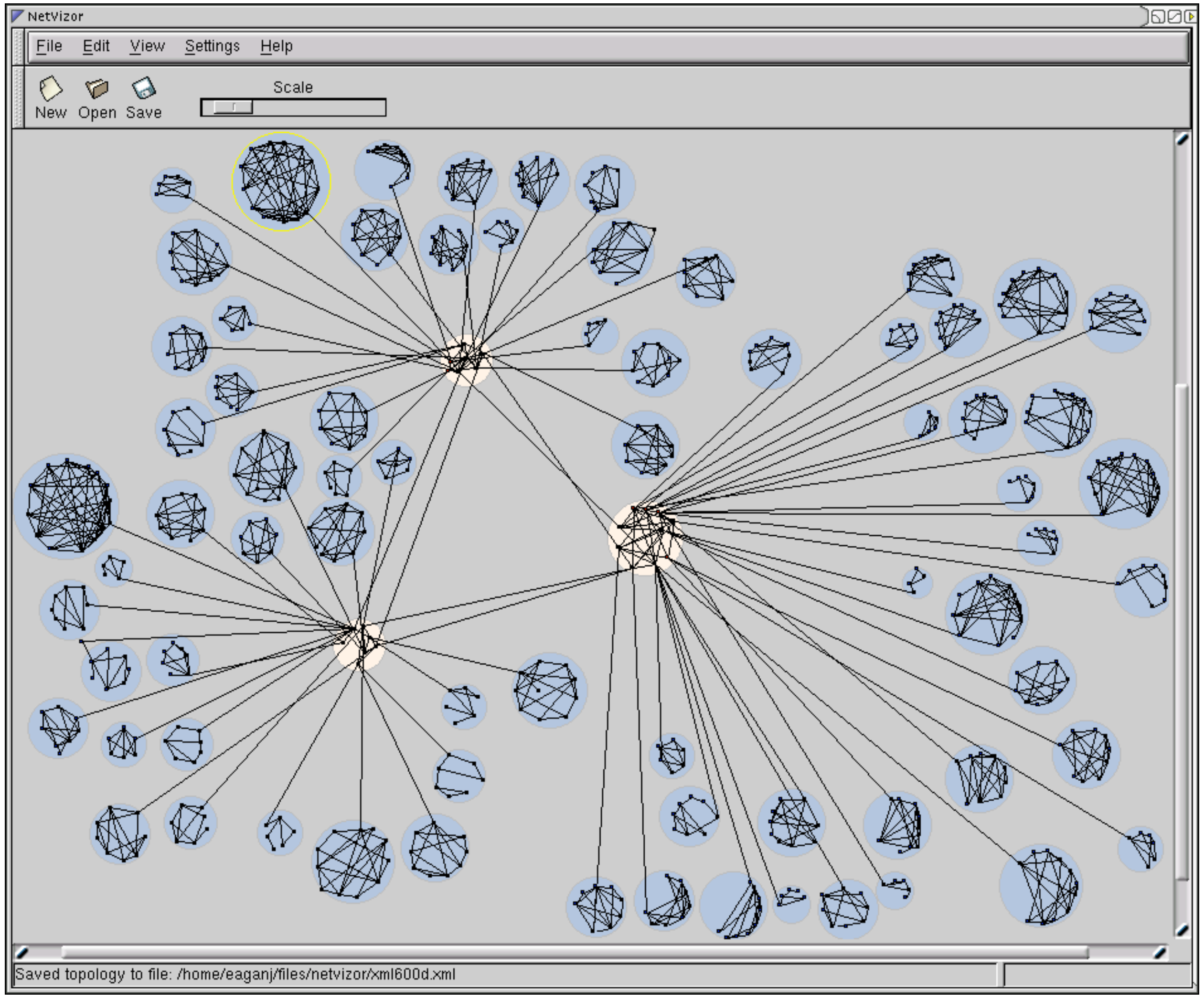


optional exercise: extend randgraph.c to Waxman

$$P(u,v) = a \exp\left(\frac{-d}{bL}\right)$$

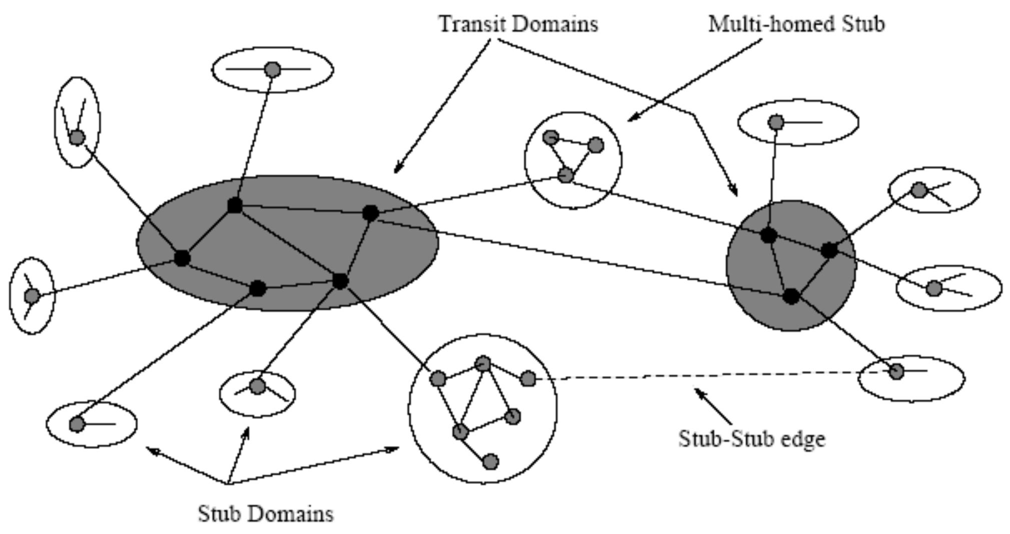
- **Caution: this is not an exponential distribution. It is a Bernoulli one (edge yes or no) where the probability of 'edge' depends on the Euclidean distance *d* of the new nodes *u* and *v*, and on the system areas diameter *L*.**
- **Parameter a adjusts the decay w.r.t. distance, b the general level of 'success'.**

I Hierarchical random graphs



Saved topology to file: /home/eaganj/files/netvizor/xml600d.xml

I Hierarchical random graphs



» Construct connected random graph: each node represents an entire transit domain

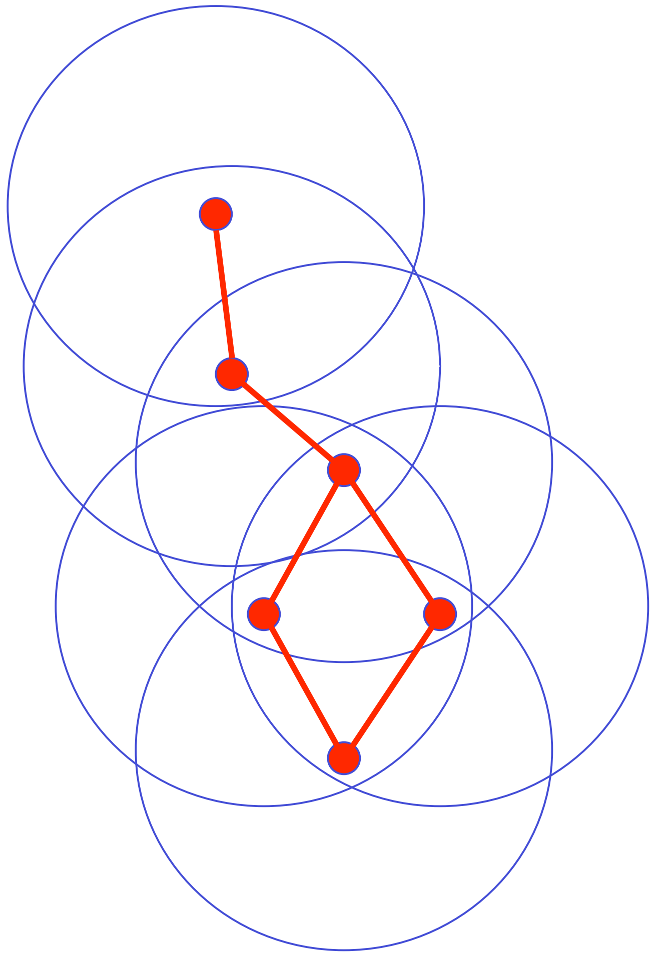
» Each node is replaced by a connected random graph representing a transit network

» For each node in transit networks: construct connected random graph representing the associated stub networks

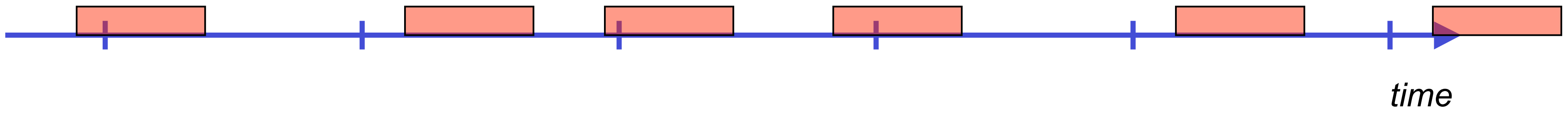
» Add some more edges between transit-and-stub and stub-and-stub networks.

I Unit disc graph (for wireless scenarios)

- For wireless scenarios – idealized ‘transmission ranges’
- There is an edge between nodes u and v if distance between nodes is smaller then a given value t.



I Constant bit-rate traffic with jitter



» Constant bit rate traffic is parameterized via
  - Rate: the sending rate or
  - Interval: interval between packets and
  - PacketSize: the constant size of the packets generated

» Introducing jitter: random flag indicating whether or not to introduce random “noise” in the scheduled departure times (default is off)

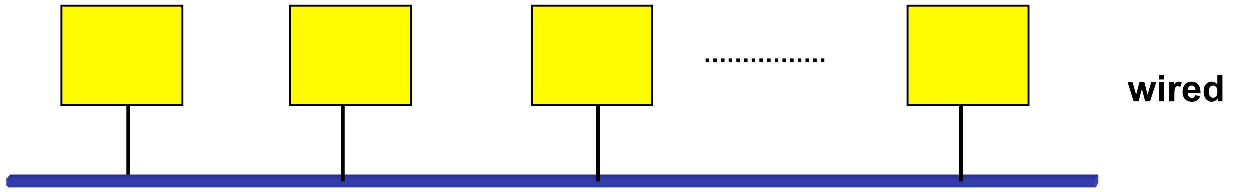
» Example:
  - packetSize 48
  - rate 64Kb
  - random 1

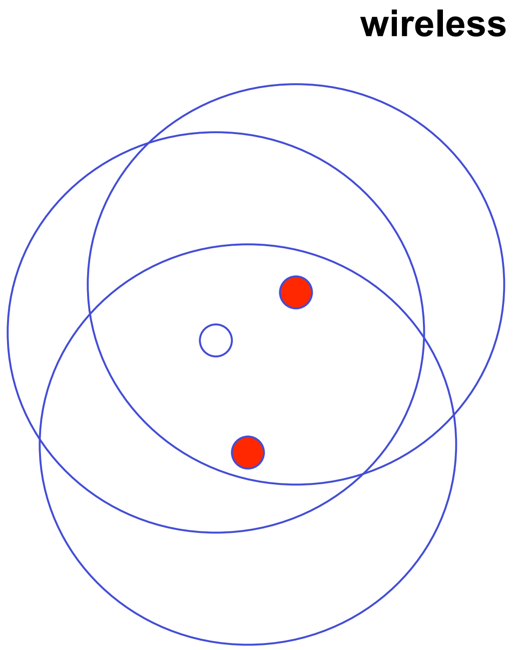
» Example (NS-2, /tools/cbr_traffic.cc):

```
double t = interval_; if (random_)

        t += interval_ * Random::uniform(-0.5, 0.5);
```

I Multiaccess communication: problem statement

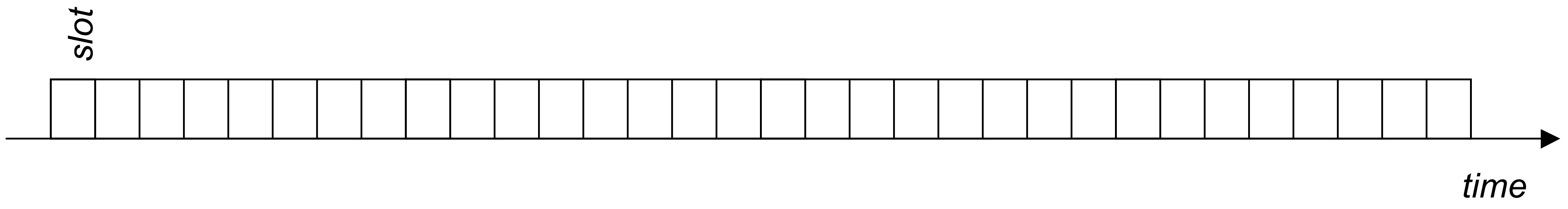
- Randomness as a mechanism for self-organization





- Assume: no centralized control
- How to organize access to shared medium?
- When two or more stations send a packet at the 'same' time, a collision occurs

I Multiaccess communication: exponential backoff

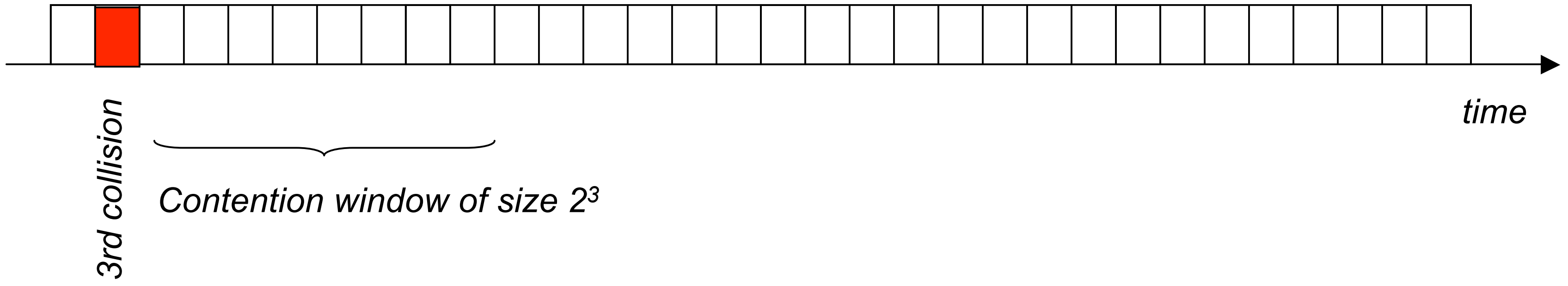


» **Binary exponential backoff:**

» **If a packet has been transmitted unsuccessfully *i* times, the transmission is rescheduled for a randomly chosen slot in the 'contention window' of size 2^i slots.**

» **Self-adjustment to number of competing stations.**

» **Example:**



I Multiaccess communication: 10Mbps Ethernet

Collision on attempt no.	Contention window	Backoff times
1	0 … 1	0 … 51.2 μs
2	0 … 3	0 … 153.6 μs
3	0 … 7	0 … 358.4 μs
4	0 … 15	0 … 768 μs
5	0 … 31	0 … 1.59 ms
6	0 … 63	0 … 3.23 ms
7	0 … 127	0 … 6.5 ms
8	0 … 255	0 … 13.1 ms
9	0 … 511	0 … 26.2 ms
10-15	0 … 1023	0 … 52.4 ms
16	N/A	Discard frame

I Multiaccess communication: NS-2 example

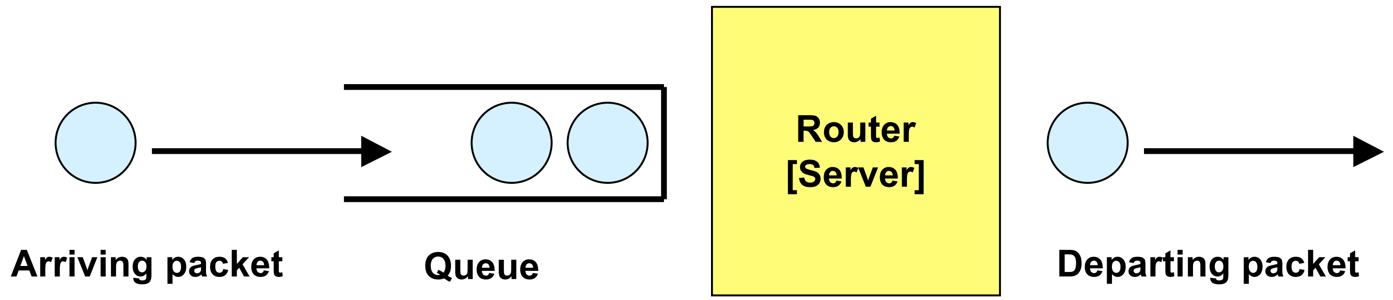
```
rtime = (Random::random() % cw) * mac->phymib_.getSlotTime();
```

- From ns-2.27/mac/mac-timers.cc
- `random()` returns random integer in [0, MAXINT]
- Efficient?
- Other approach starting with U(0,1):

$$\lfloor (cw + 1)U \rfloor$$

- Exercise: Experiment with randspeed2 (which is fastest?)

II Generating other random variates: introduction



How to generate exponentially distributed inter-arrival times?

- » **Algorithms to produce observations ("variates") from some desired input distribution (exponential, gamma, etc.)**
- » **Formal algorithm - depends on desired distribution.**
- » **But *all* algorithms have the same general form:**
  - – **Generate one or more IID U(0, 1) random numbers**
  - – **Transformation (depends on desired distribution)**
  - – **Return *X* ~ desired distribution**
- » **Note critical importance of a good random-number generator**

II Criteria

May be several algorithms for a desired input distribution form; want:

- **Exact: *X* has *exactly* (not approximately) the desired distribution**
  - **Example of approximate algorithm:**
    - **Treat $Z = 6 - U1 + U2 + ... + U12$ as normal distribution N(0,1)**
    - **Mean, variance correct; rely on Central Limit Theorem for *approximate* normality**
    - **Range clearly incorrect**
- **Efficient:**
  - **Low storage**
  - **Fast (marginal, setup)**
  - **Efficient regardless of parameter values (*robust*)**
- **Simple: Understand, implement (often tradeoff against efficiency)**

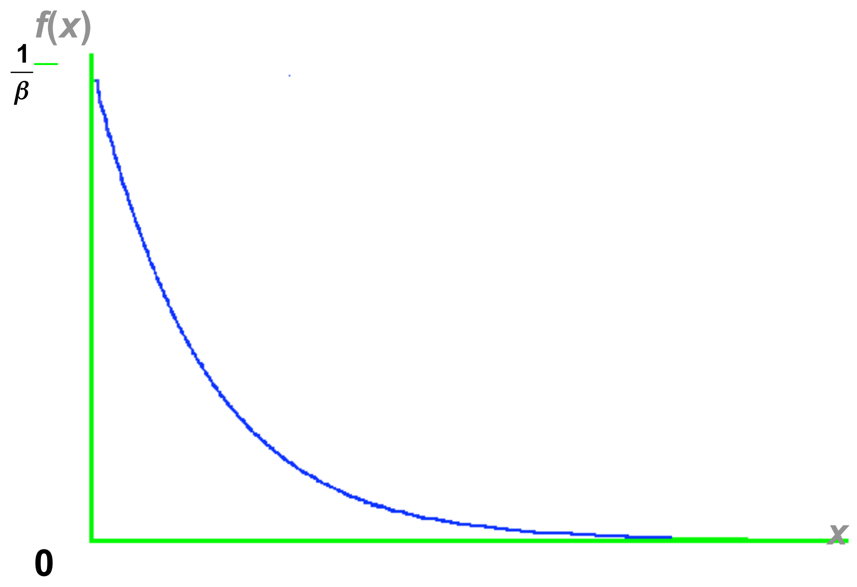
II Recap: distribution function

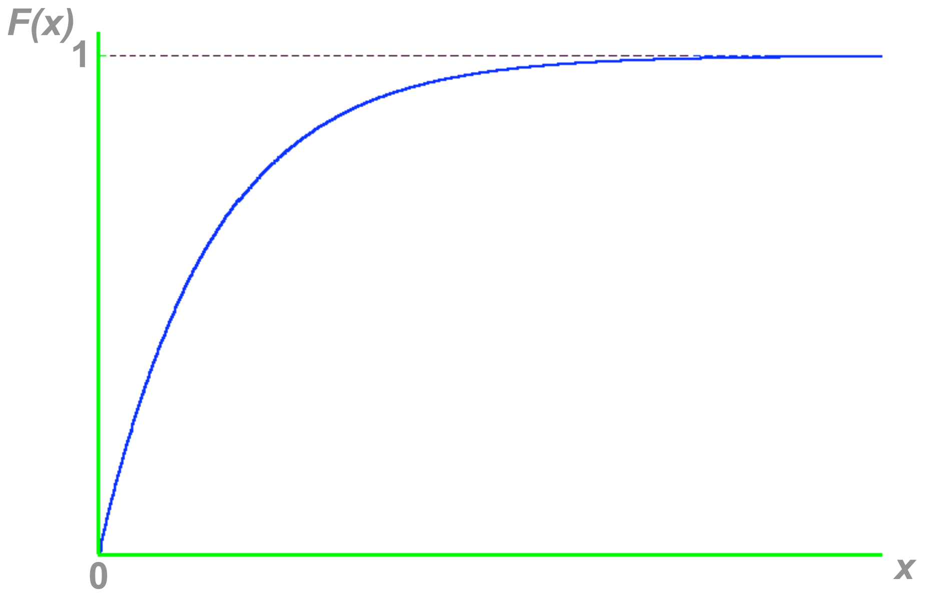
Definition:

the (cumulative) distribution function F(x) for a random variable

X: Ω ! R is defined as

$$F(x) = P(X \leq x) \quad \text{for} \quad -\infty < x < \infty$$





II Recap: strictly increasing function

» A distribution function F is *strictly* increasing when

$$x_1 < x_2 \rightarrow F(x_1) < F(x_2)$$

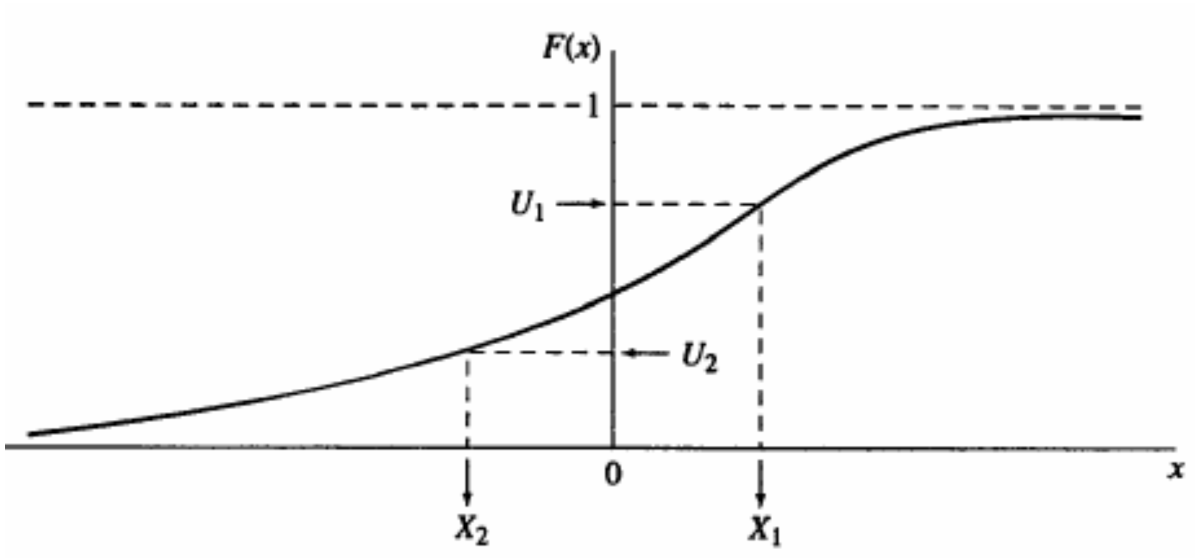
» For a distribution function F that is continuous and strictly increasing when $0 < F(x) < 1$, the inverse function F^{-1} exists on (0,1).

II Inverse transform method

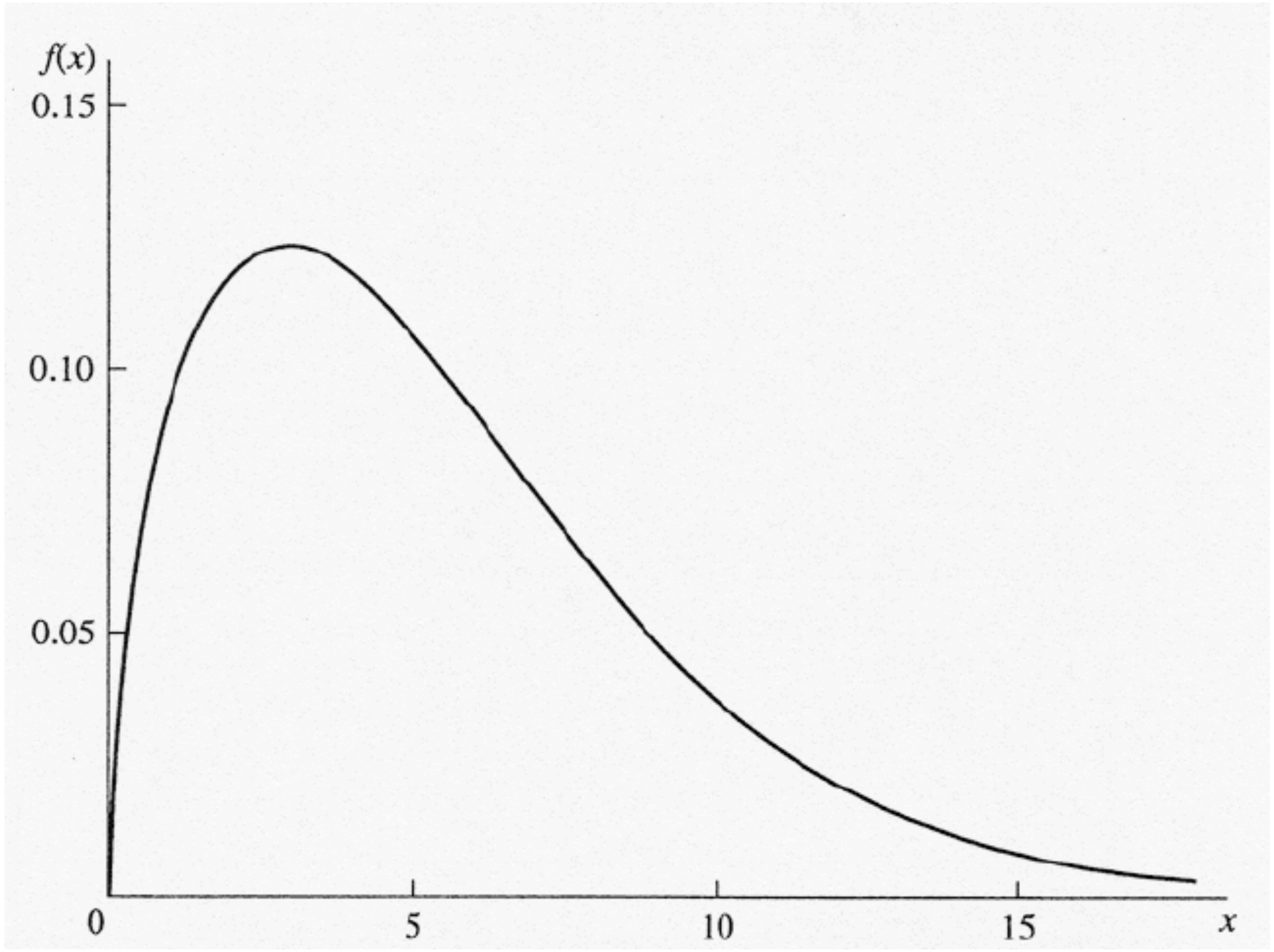
» Goal: generate a random variate X that is continuous and has distribution function F that is continuous and strictly increasing when 0 < F(x) < 1.

» Strategy: Inverse transform algorithm

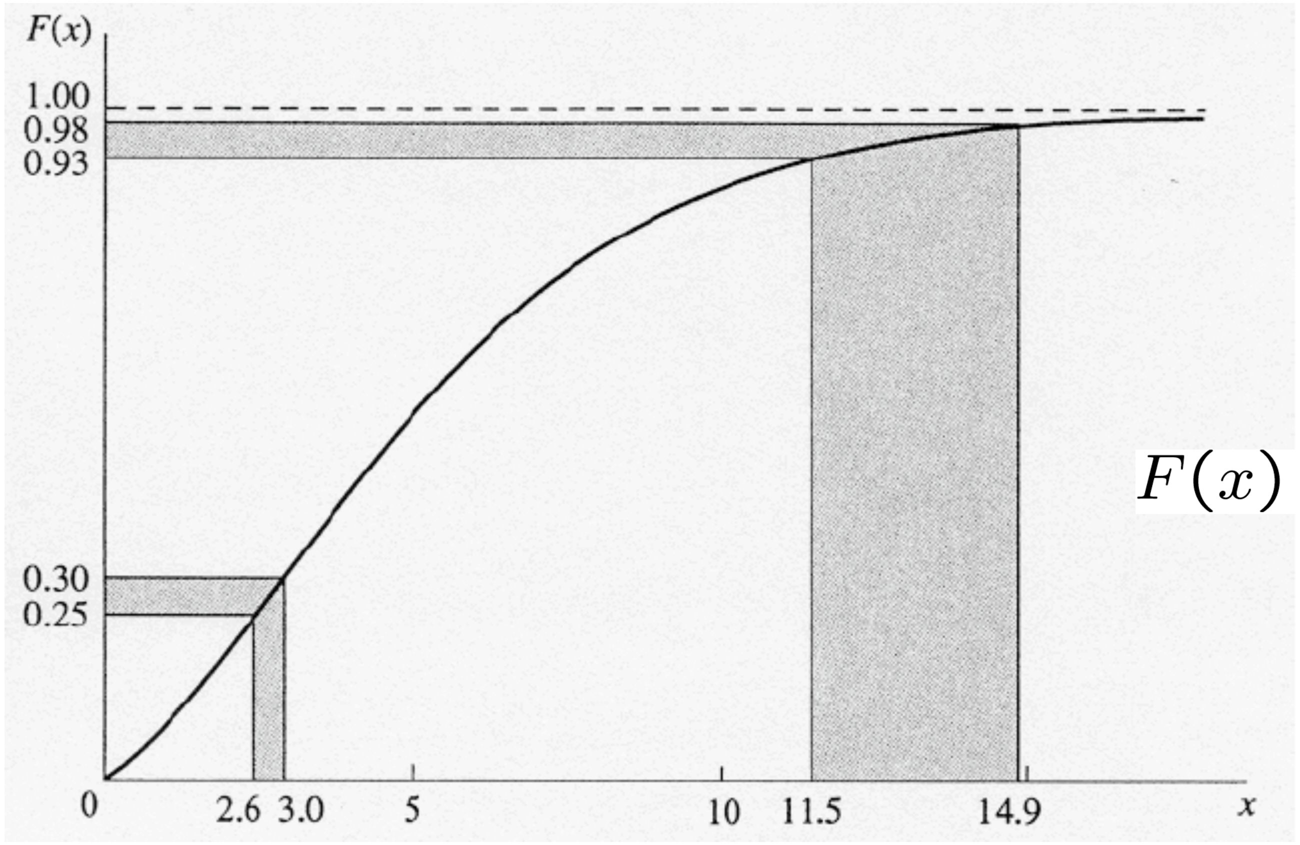
1. Generate U ~ U(0,1)
2. Return X = $F^{-1}(U)$



II Intuitive explanation I: density

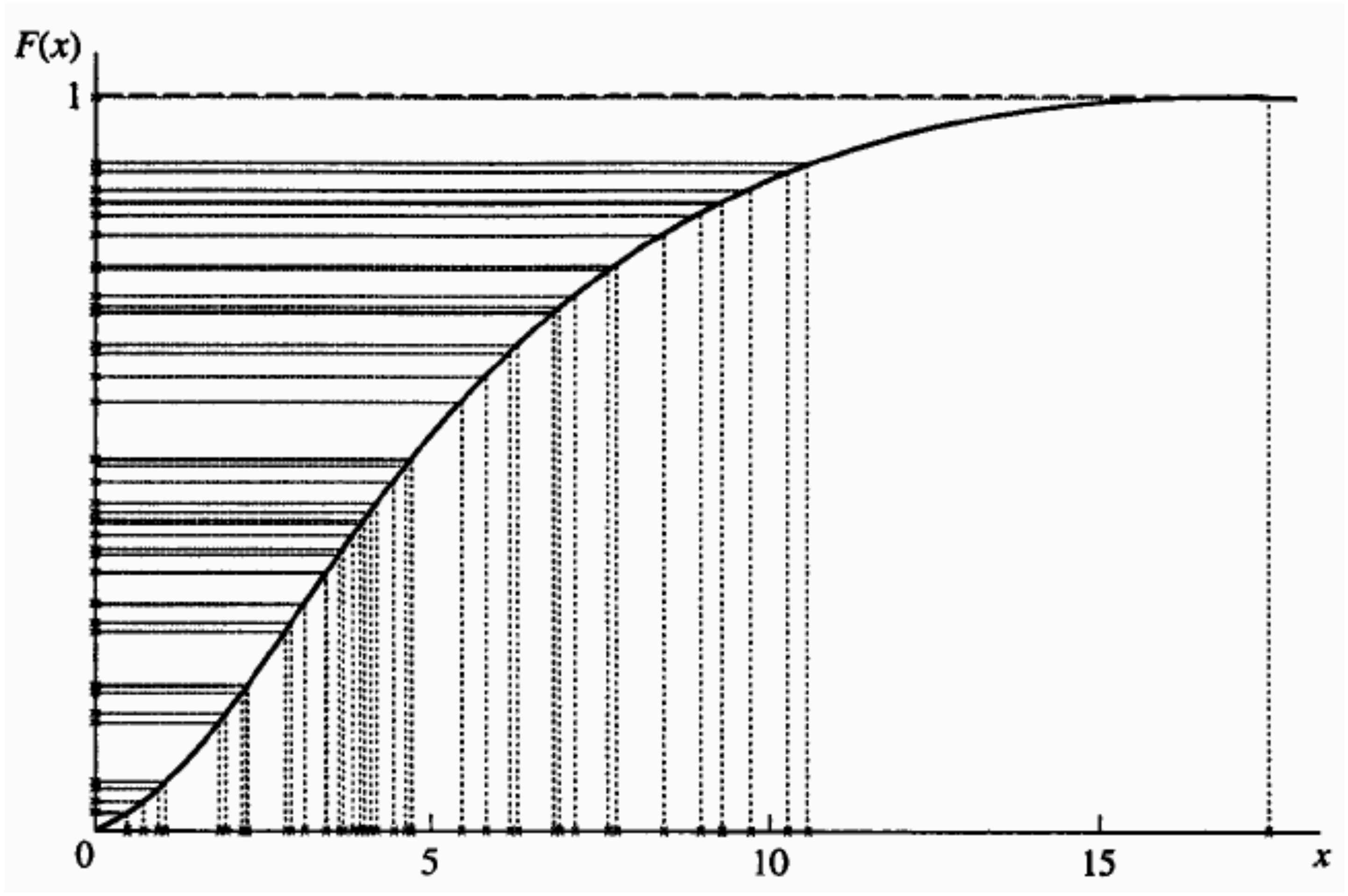


II Intuitive explanation II: distribution function



$$F(x) = P(X \leq x)$$

II Intuitive explanation III: algorithm at work (50 samples)



II Proof

$$P(X \leq x) = P(F^{-1}(U) \leq x)$$

» **Definition**

$$P(F^{-1}(U) \leq x) = P(U \leq F(x))$$

» **Properties of F**

$$P(U \leq F(x)) = F(x)$$

» **Uniform distribution, definition of distribution function**

$$P(X \leq x) = F(x)$$

II Example: exponential distribution

- **Density**

$$f(x) = \begin{cases} \frac{1}{\beta} e^{-x/\beta} & \text{if} \quad x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

- **Distribution function**

$$F(x) = \begin{cases} 1 - e^{-x/\beta} & \text{if} \quad x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

- **Inverse transform**

$$F^{-1}(u) = -\beta \ln(1 - \underset{\substack{\uparrow \\ \textbf{uniformly distributed}}}{u})$$

II Exponential distribution: C code

From M/M/1 example:

```
float expon(float mean)   /* Exponential variate
   generation function. */

{/* Return an exponential random variate with mean
   "mean". */

     return -mean * log(lcgrand(1));}
```

From NS-2 (tools/rng.h):

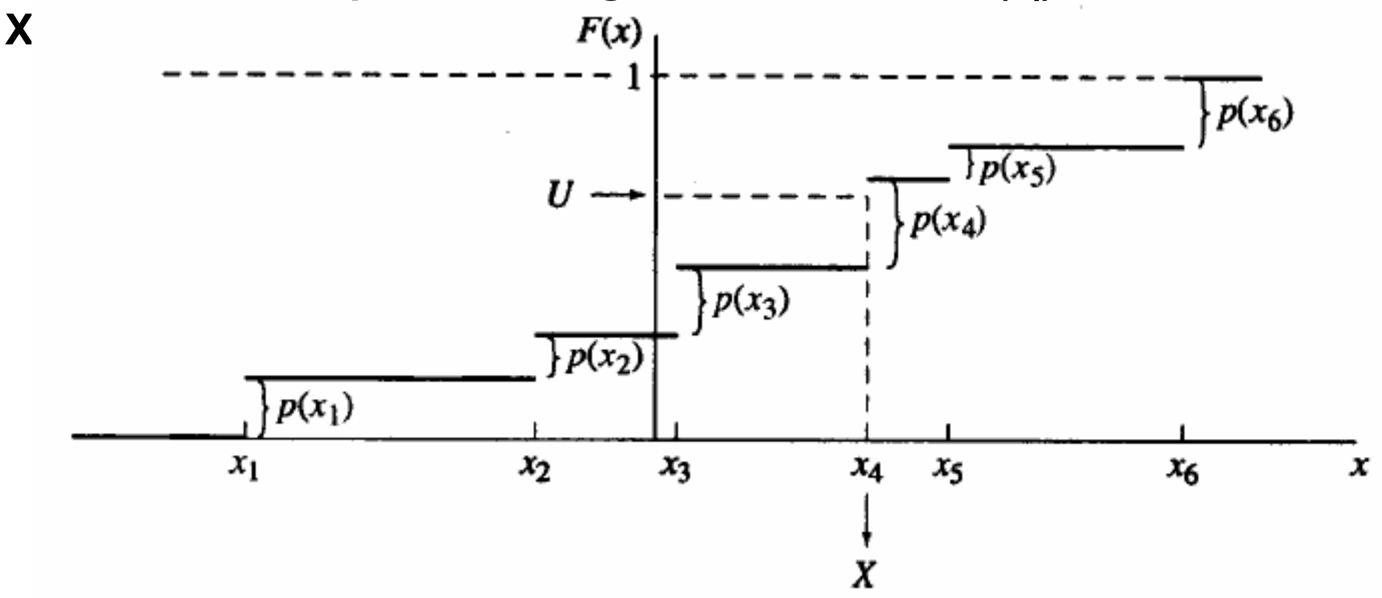
```
inline double exponential ()

   {return (-log(uniform()));}

inline double exponential (double r)

   {return (r * exponential());}
```

II Inverse transform method for discrete distributions

- Assume: random variable X can take only values x_1, x_2, … with $x_1 < x_2 <$ …
- Probability mass function: $p(x_i) = P(X = x_i)$
- Distribution function: $F(x) = P(x \cdot x) = \sum_{x_i \cdot x} p(x_i)$

Algorithm:

1. Generate U ~ U(0,1)
2. Determine the smallest positive integer I such that $U \cdot F(x_I)$, and return X



II Discrete case: efficiency considerations

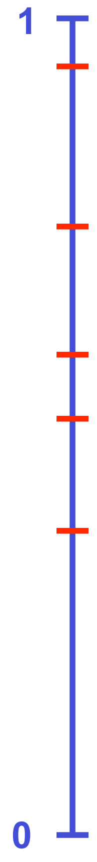
Step 2 involves a "search" of some kind; several computational options:

- **Direct left-to-right search—if $p(x_i)$'s fairly constant**
- **If $p(x_i)$'s vary a lot, first sort into decreasing order, look at biggest one first, ..., smallest one last—more likely to terminate quickly**
- **Example:**

  Probabilities {p_1=1/8, p_2=1/8, p_3=1/4, p_4=1/2}.

  A: order 1, 2, 3, 4. Expected number of comparisons: 11/8

  B: order 4, 3, 2, 1. Expected number: 9/8
- **Exploit special properties of the form of the $p(x_i)$'s, if possible**
  - **To facilitate computation of F(x)**



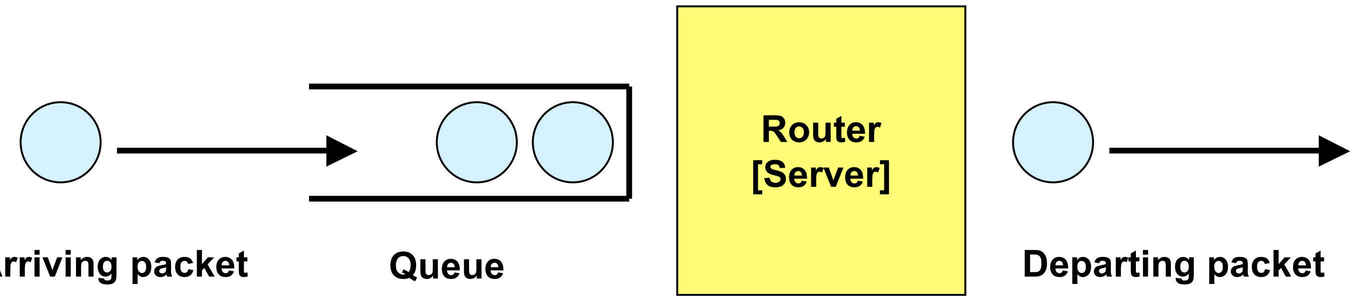
II Generalizations and limits

» **Generalized Inverse-Transform Method**
  - **Valid for *any* CDF $F(x)$: return $X = \min\{x: F(x) \geq U\}$, where $U \sim$ U(0,1)**
    - **Continuous, possibly with flat spots (i.e., not strictly increasing)**
    - **Discrete**
    - **Mixed continuous-discrete**

» **Problems with Inverse-Transform Approach**
  - **Must invert CDF, which may be difficult (numerical methods)**
  - **May not be the fastest or simplest approach for a given distribution**

III Modeling examples

- … using exponential variates
- Recap: Inter-arrival times



- Examples for topologies
  - Mobile case: random direction mobility
- Examples for synthetic data traffic
  - On/off sources for generating bursty traffic

III Random direction mobility

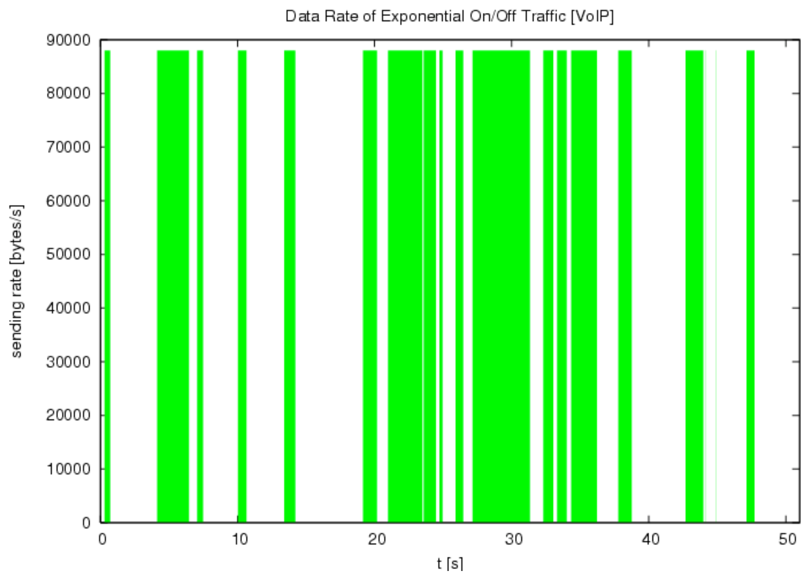
- Last lecture: random waypoint mobility
- Various flavors around
- Node chooses random direction sampled from uniform distribution over $[0, 2\pi)$
- Node chooses speed either from uniform or normal distributions
- Node changes direction/speed after a time interval sampled from an exponential distribution
- At system area’s boundary: wrap-up, reflection …

III On/off sources for modeling bursty traffic

- NS-2: EXPOO_Traffic—generates traffic according to an Exponential On/Off distribution. Packets are sent at a fixed rate during on periods, and no packets are sent during off periods. Both on and off periods are taken from an exponential. Packets are constant size.
- packetSize_ the constant size of the packets generated
- burst_time_ the average “on” time for the generator
- idle_time_ the average “off” time for the generator
- rate_ the sending rate during “on” times
- Example:
  - packetSize_ 210
  - burst_time_ 500ms
  - idle_time_ 500ms
  - rate_ 100k

III VoIP modeling

- ITU-T recommendations for VoIP modeling:
- On periods: exponential with mean 1.004s
- Off periods: exponential with mean 1.587s
- Sending rate during on periods: 88kbps



Summary / Educational Goals

- » Examples of how to model network topologies and data traffic
- » Discussion on how to generate ‘arbitrary’ random variates
- » Inverse transform method
  - – Particularly good for exponential variates

So far:
- • Generation of uniform, discrete and exponential variates
- • Random topologies (static, wireless, mobile networks),
  Synthetic data traffic (CBR, exponentially distributed
  inter-arrival times, on/off-sources)

References

» **Exponential backoff**

- **Charles E. Spurgeon, *Ethernet – The Definite Guide*, O'Reilly, 2000; pp. 53-60**
- **Dimitri Bertsekas, Robert Gallager, *Data Networks*, 2nd ed., Prentice Hall, 1992, pp. 286**

» **Inverse transform method**

- **Averill M. Law, W. David Kelton, *Simulation Modeling and Analysis*, 3rd ed., McGrawHill, 2000**

» **Waxman model**

- **Bernard M Waxman Routing of multipoint connections IEEE Journal on Selected Areas in Communications, 6 (9): 1617-1622, 1988.**

» **Hierarchical random graphs**

- **Geogia Tech Internet Topology Map (GT-ITM), *models.ps***