CS630 Graduate Algorithms

October 24, 2024

by Dora Erdos and Jeffrey Considine

Contention in distributed systems:

- randomized load balancing KT ch. 13.9
  - random variables and expectation KT ch. 13.3
  - Chernoff bounds KT ch. 13.8
- packet routing KT ch. 13.10

slides by Dora Erdos and Kevin Wayne

Load balancing

Problem: System in which m jobs arrive in a stream and need to be processed immediately on n identical processors. Find an assignment that balances the workload across processors.

Centralized controller. Assign jobs in round-robin manner. Each processor receives at most $\lceil m / n \rceil$ jobs.

Decentralized controller. Assign jobs to processors uniformly at random.

- How likely is it that the load on each processor balances out?
- How likely is it that some processor is assigned "too many" jobs?

Random experiments and the sample space

Random experiment:

- a process that produces uncertain outcomes from a well-defined sample space
- outcome: the result of an experiment (e.g. the value on the dice)
- sample space: the set of all possible outcomes
- event: any subset of the sample space

example: flip a coin, the outcome is heads or tails

Random Variables

A random variable X is a real valued *function* that assigns a *numerical value* to each possible outcome of a random experiment.

$$X : S \to \mathbb{R}$$

example. For your experiment you flip a coin five times and record the sequence of heads and tails. Then the sample space S consists of $2^5 = 32$ elements.

$$S = \{HHHHH, HHHHT, \ldots, TTTTT\}$$

The probability of each of these outcomes is $\frac{1}{2^5}$

Let X be the function that counts the number of Hs in the outcome, e.g. *X(HTTHH) = 3*

The range of X is *Range(X)* = R_X= *{0,1, 2, 3, 4, 5}*

Pr(X=3) = $\dfrac{\binom{5}{3}}{2^5}$

$\binom{5}{3}$ ← 5 possible coin flips, 3 "locations" to put H

2^5 ← all possible outcomes

Discrete Random Variables

example 1: Flip a coin five times. X is the number of heads in the outcome. The range is $R_X = \{0, 1, 2, 3, 4, 5\}$

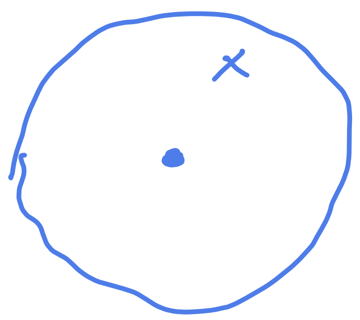
example 2: Y is the number of flips of a coin before heads appears for the first time. $R_Y = \{1, 2, 3, \ldots\}$

When the range is finite or countable infinite it is called a discrete random variable.

Continuous Random Variables

example 3: Z is the time until the next request to a server arrives. Note that time is continuous and so is the value of Z, $R_Z = [0,\infty)$

example 4: W is the distance of a thrown dart from the center of a 1 meter radius board, $R_W = [0,1)$.



Z and W are called continuous random variables.

Probability Mass Function of a Discrete Rnd. Var.

Let X be a discrete random variable. Its Probability Mass Function (PMF) P_X assigns a probability to each number in the range of X.

$$P_X : R_X \rightarrow [0,1]$$

X in the subscript is to clarify that this is the probability corresponding to random variable X. most often it's clear what it refers to and we omit it from the notation.

The value $P_X(X = a)$ or $P_X(a)$ is the probability that the random variable X takes on the value a.

P_X is a probability measure, that is it has the properties

$$P_X(X = a) \geq 0$$

$$\sum_{a \in R_X} P_X(X = a) = 1$$

Probability Mass Function of a Discrete Rnd. Var.

example 1: X is the outcome of the role of a dice. What is its range and PMF?
(list the possible outcomes and their probabilities)

$$R_X = \{1, 2, 3, 4, 5, 6\}$$

$$P(X = a) = \frac{1}{6}$$

example 2: We role a dice, the random variable is $X = \begin{cases} 1 & roll = \{1,2\} \\ 0 & roll = \{3,4,5,6\} \end{cases}$

$$R_X = \{0, 1\}$$

$$P(X = 1) = \frac{2}{6} = \frac{1}{3}$$

$$P(X = 0) = 1 - P(X = 1) = \frac{2}{3}$$

example 3: Y is the number of times we flip a coin until the first heads appears. What is its range and PMF?

TopHat - range

Random variable Y is the number of times we flip an *unbiased* coin until the first heads appears.

$p(\text{Heads}) = \frac{1}{2}$

What is its range?

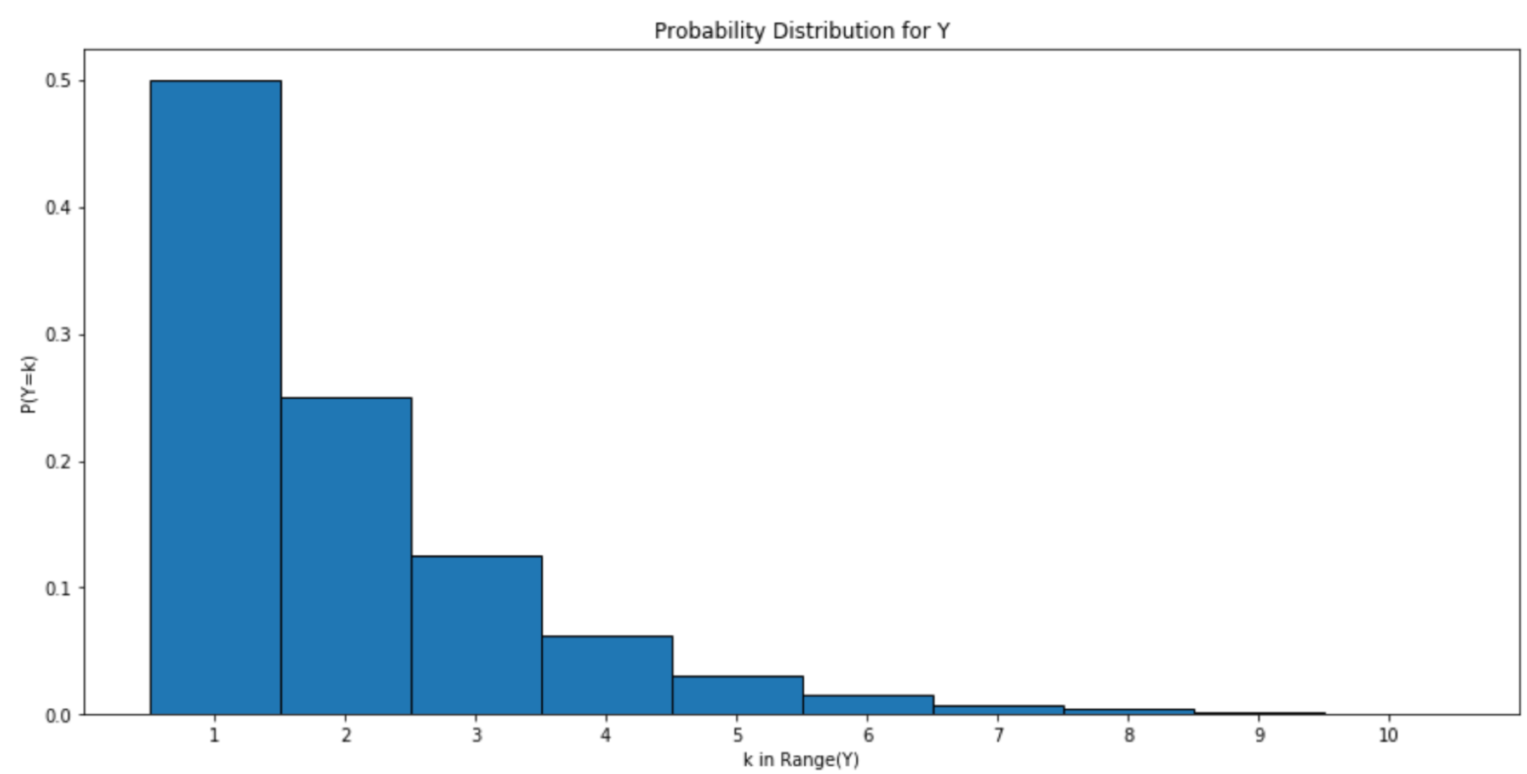
A. $R_Y = \{0,1\}$

B. $R_Y = \{1,2,3,4,5,6\}$

C. $R_Y = \{1,2,3,\ldots\infty\}$

D. $R_Y = \frac{1}{2}$

E. $R_Y = \left(\frac{1}{2}\right)^k$



TopHat - PMF

Random variable Y is the number of times we flip an *unbiased* coin until the first heads appears. Its range is $R_Y = \{1,2,3,\ldots\infty\}$

$P(H) = \frac{1}{2}$ $P(T) = \frac{1}{2}$

success

What is its PMF?

For $k = 1,2\ldots\infty$ we have

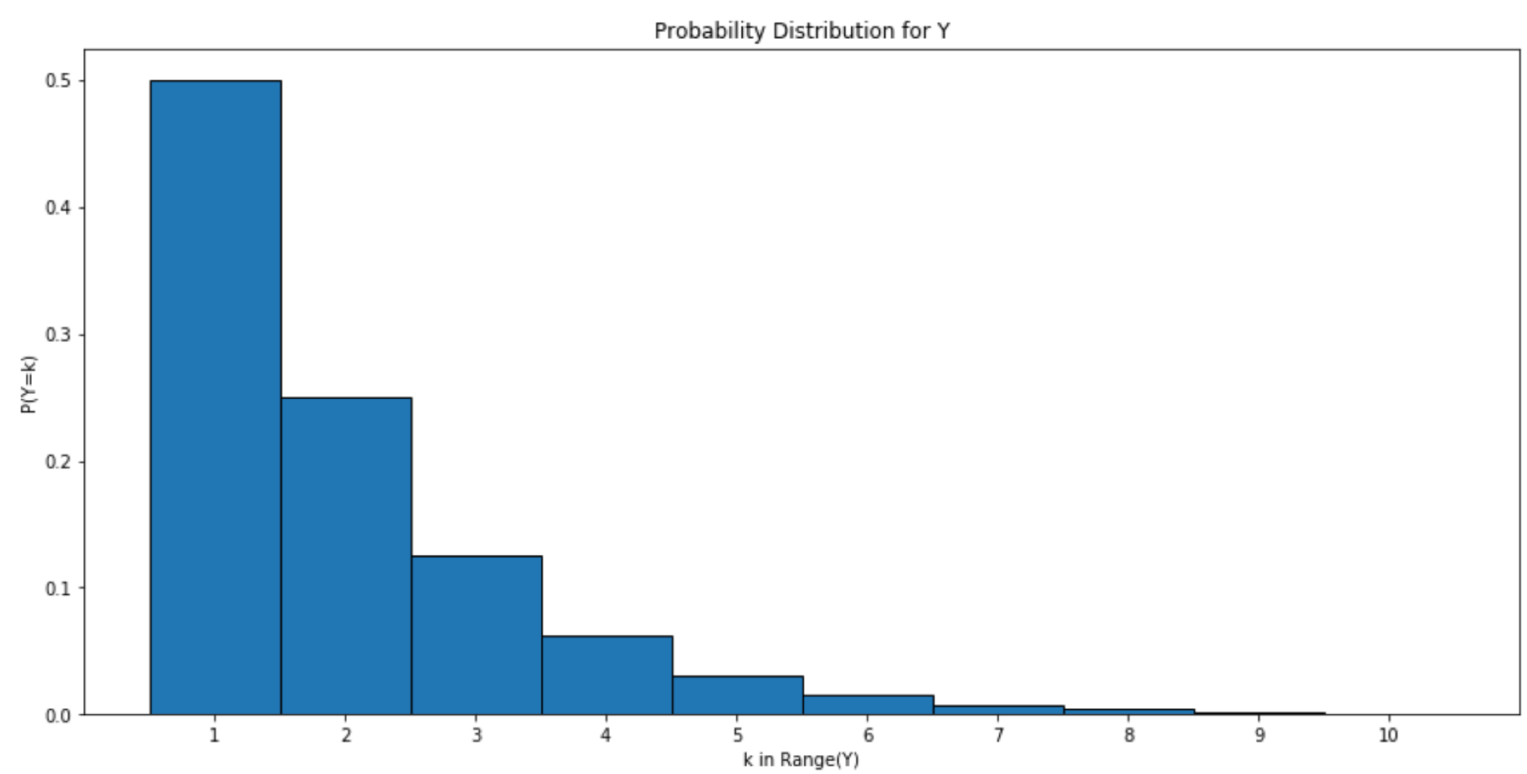
A. $P(Y=k) = \begin{cases} 0 & \textit{successful} \\ 1 & \textit{else} \end{cases}$

B. $P(Y=k) = k$

C. $PY = k) = \dfrac{1}{2}$

D. $P(Y=k) = \left(\dfrac{1}{2}\right)^k$

unsuccessful $\left(\frac{1}{2}\right)^{k-1} \cdot \frac{1}{2}$



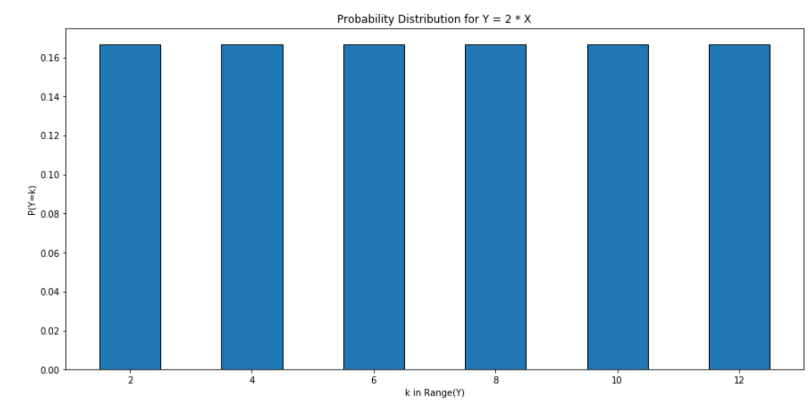
Random variables as function of other random variables

We can describe a function of a random variable to create a new one.

Let X be the outcome of a *role of the dice*. Exercise: compute the range and pmf of the following variables.

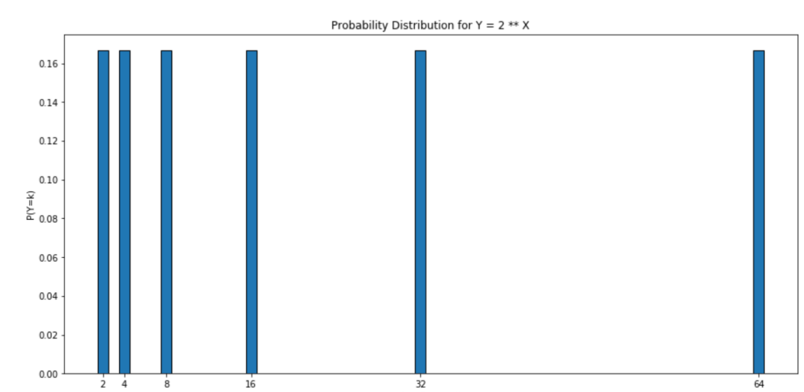
$$Y = 2X$$

$R_Y = \{2, 4, 6, 8, 10, 12\}$



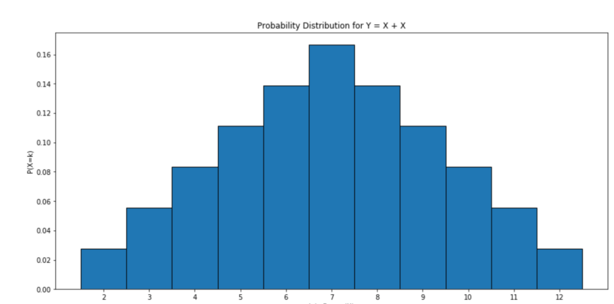
$$Y = 2^X$$

$R_Y = \{2^1, 2^2, 2^3 \ldots 2^6\}$



$$Y = X + X$$

Note that the outcome of the two dice rolls can be different.



TopHat - combination of random variables

We can describe a function of a random variable to create a new one.

Let X be the outcome of a *role of the dice,* $Y = X + X$. What is the range of Y?

(Note that the outcome of the two dice rolls can be different.)

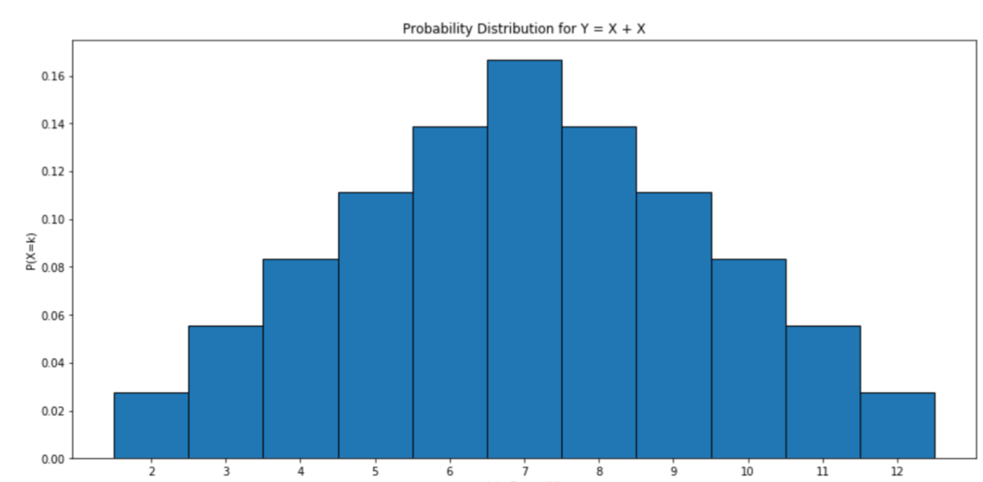
$R_X = \{1,2,3,4,5,6\}$

$X_1 = 1 \quad X_2 = 1$

A. $R_Y = \{1,2,3,4,5,6\}$

B. $R_Y = \{2,4,6,8,10,12\}$

C. $R_Y = \{2,3,\ldots,11,12\}$ ← sum of rolling two dice



Independence and Conditional of Random Variables

We can define the independence of two random variables the same way as we did for the probability function. Discrete random variables X and Y are independent if

$$P(X = k, Y = \ell) = P(X = k)P(Y = \ell)$$

We can also define conditional variables.

$$P(X = k \mid Y = \ell) = \frac{P(X = k, Y = \ell)}{P(Y = \ell)}$$

intuitively: what fraction of the time that $Y = \ell$ does X=k happen.

Reminder: Bayes' rule for conditional probability of events $P(A \mid B) = \dfrac{P(A \cap B)}{P(B)}$

Cumulative Distribution Function

discrete

The Cumulative Distribution Function (CDF) of a random variable X is the probability of X taking on a range of values.

for continuous rand vars take the $\int$ instead of sum.

$$F_X(k) = P(X \leq k) = \sum_{a \leq k} P(X = a)$$

prob of $X \leq$

hope that this prob is high

$$P(\ell \leq X \leq k) = P(X \leq k) - P(X \leq \ell - 1)$$

Markov's inequality:

$$P(X \geq \alpha) \leq \frac{E[X]}{\alpha}$$

$\hookrightarrow$ X is $\geq$ than some value

hope that this prob is low

Bernoulli trial and common distributions

Bernoulli trial:
flip a coin that is heads with probability *p.* The random variable X is 1 if it is heads and 0 otherwise.

$$R_X = \{0, 1\}$$

$$P_X(1) = p,\ P_X(0) = 1 - p$$

Many random processes are some kind of combination of Bernoulli trials.

Binomial Distribution: Count the number of successes in *n* independent trials

Geometric Distribution: The number of trials until the first success

Binomial Distribution

Count the number of successes in n independent Bernoulli trials. Let $Y \sim Bernoulli(p)$

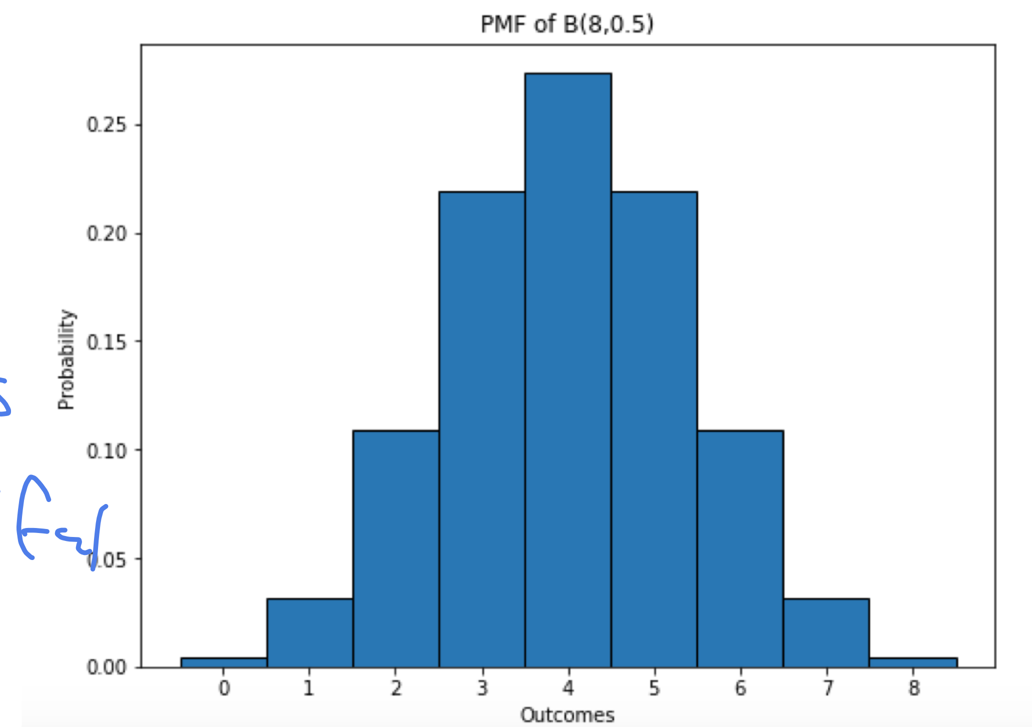
$$(a+b)^2 = a^2 + 2ab + b^2$$
$$= a^2 b^0 + \binom{2}{1} a^1 b^1 + \binom{2}{2} a^0 b^2$$

$X =$ number of successes in n trials $= Y + Y + \ldots Y$

$R_X = \{0, 1, 2, \ldots n\}$

$P\left(X = k\right) = \binom{n}{k} p^k (1-p)^{n-k}$

number of ways to select which of the trials is successful



In notation $X \sim Binom(n, p)$

Is this P a proper PDF?

$P(X = k) \geq 0$

compute $\sum_{k=0}^{n} P(X = k)? = 1$

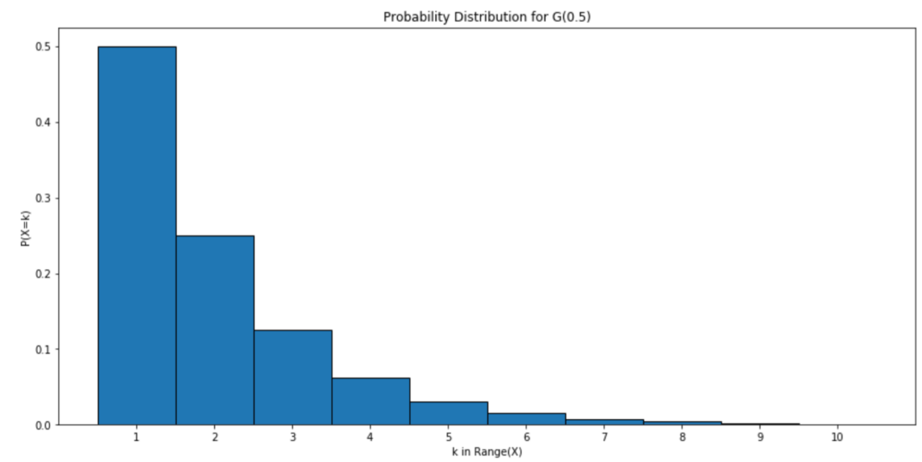
$$\sum_{k=0}^{n} \binom{n}{k} p^k (1-p)^{n-k} = (p + (1-p))^n = 1^n$$

expand this

Geometric Distribution

X is the number of independent identical Bernoulli trials until the first success.

$$X \sim Geom(p)$$



$$R_X = \{1, 2, \ldots\}$$

$$P\left(X = k\right) = (1-p)^{k-1}p$$

This really is a probability distribution as it sums up to 1!

$$a = \sum_{k=1}^{\infty} P(X=k) = p + (1-p)p + (1-p)^2 p + \ldots =$$

$$p + (1-p)\big(p + (1-p)p + (1-p)^2 p + \ldots\big) = p + (1-p)\left(\sum_{k=1}^{\infty} P(X=k)\right) = p + (1-p)a$$

Expected Value

discrete

Let X be some random variable. Before performing the trial what do we expect the outcome to be?

For discrete random variables the expected value or mean is the weighted average of the range of X.

$$E[X] = \sum_{a \in R_X} a \cdot P(X = a) \quad = \text{weighted average}$$

Linearity of expectation: let X and Y be two discrete random variables and a and b be constants. Then

$$E[aX + b] = aE[X] + b$$

$$E[X + Y] = E[X] + E[Y]$$

exercise: prove this by writing out the definition.

Expected Value

Find the expected value if

$$X \sim Bernoulli(p)$$

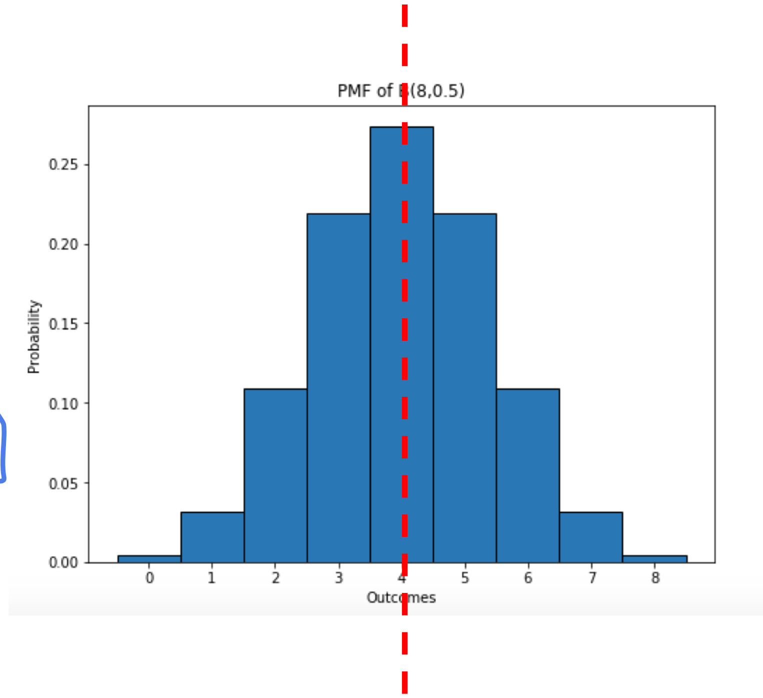
$E[x] = 1 \cdot p + 0(1-p) = p$

$$X \sim Binom(n, p)$$

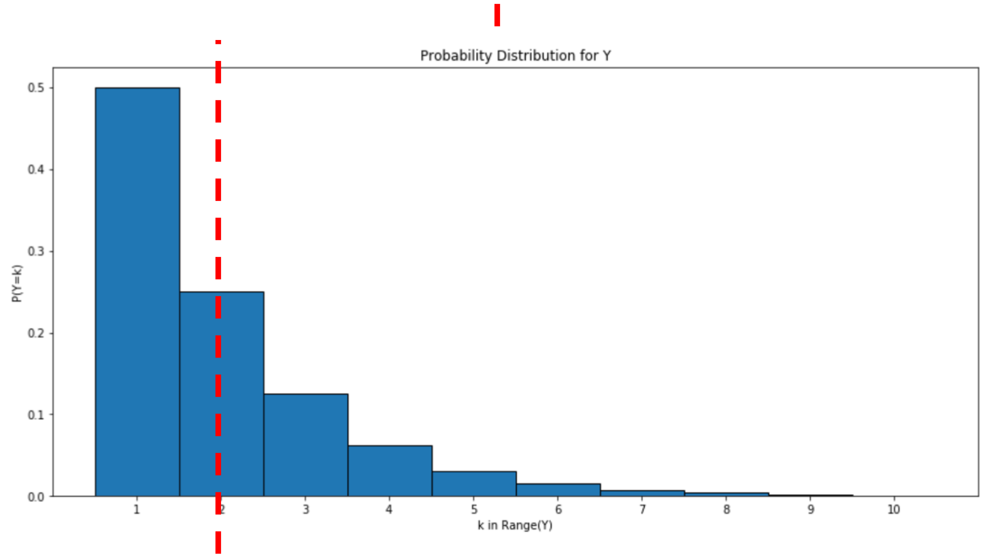
n indep trials

$E[Binom] = E[x] + E[x] + \ldots + E[x] = n \cdot p$

linearity of exp.



$$X \sim Geom(p)$$



~~TopHat - Expected Value~~

Find the expected value if X is Binom(n,p) (= number of heads in case of n biased coin flips with probability p)

Each coin flip is an independent trial, hence we can use linearity of expectation

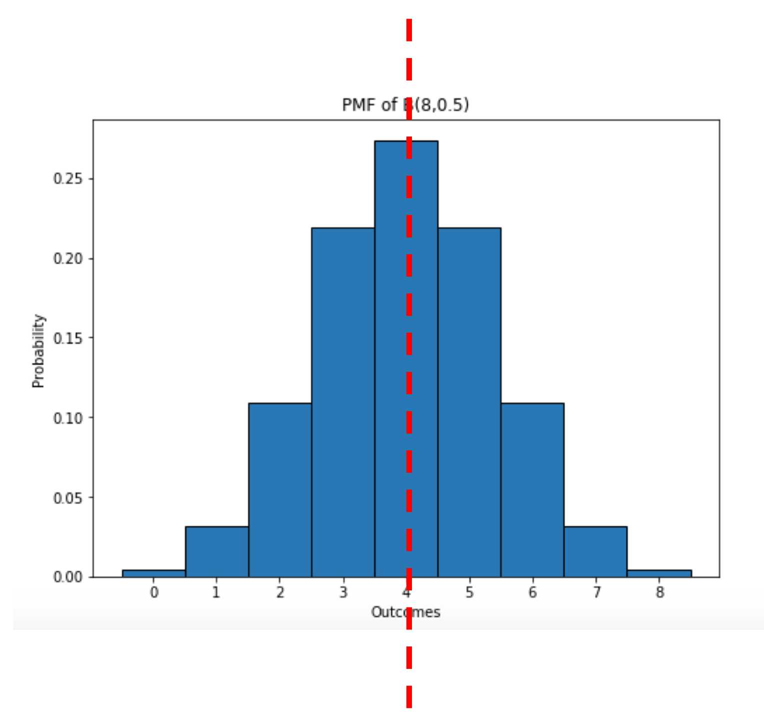
E[Binom(n,p)] = E[X]+E[X]+….+E[X] = ?

A. E[Binom(n,p)] = p

B. E[Binom(n,p)] = k

C. E[Binom(n,p)] = n

D. E[Binom(n,p)] = np



Expected Value

Find E[X] if X is

$$X \sim Bernoulli(p)$$

$$E[X] = 1 \cdot p + 0 \cdot (1-p) = p$$

$$X \sim Geom(p)$$

$$E[X] = \sum_{k=1}^{\infty} k(1-p)^{k-1} p =$$

$$= p \sum_{k=1}^{\infty} k(1-p)^{k-1} =$$

$$= p \frac{1}{(1-(1-p))^2} = \frac{1}{p}$$

trick: $\sum_{r=0}^{\infty} x^r = \frac{1}{1-x}$ for $0 < x \leq 1$

$$\frac{d}{dx} \sum_{r=0}^{\infty} x^r = \frac{d}{dx} \frac{1}{x}$$

$$\sum_{r=0}^{\infty} r x^{r-1} = \frac{1}{(1-x)^2}$$

Back to random load balancing

Random Load balancing - is it balanced?

Suppose for now that n = m

- Let X_i = number of jobs assigned to processor i. random var
  - ideally each processor would have one job
- Let $Y_{ij} = 1$ if job j assigned to processor i, and 0 otherwise. rnd var
- We have $\mathrm{E}[Y_{ij}] = 1/\mathrm{n}$.
  - $\mathrm{Y_{ij}}$ is sometimes called an indicator
- Thus, $X_i = \sum_j Y_{ij}$, and $\mu = \mathrm{E}[X_i] = 1$.

$E[X_i] = E\left[\sum_{j=1}^{n} Y_{ij}\right] = E[Y_{i1}] + E[Y_{i2}] + \dots = n\frac{1}{n} = 1$

Question. What is the probability that X_i is large? Thus, more than 1 job is assigned to processor i.

- $\Pr(X_i > c)$ for some c?

prob of "bad" out come

prob. of bad luck

happy if X_i is a low num.

$P(X_i \le c) \rightarrow$ prob of good outcome

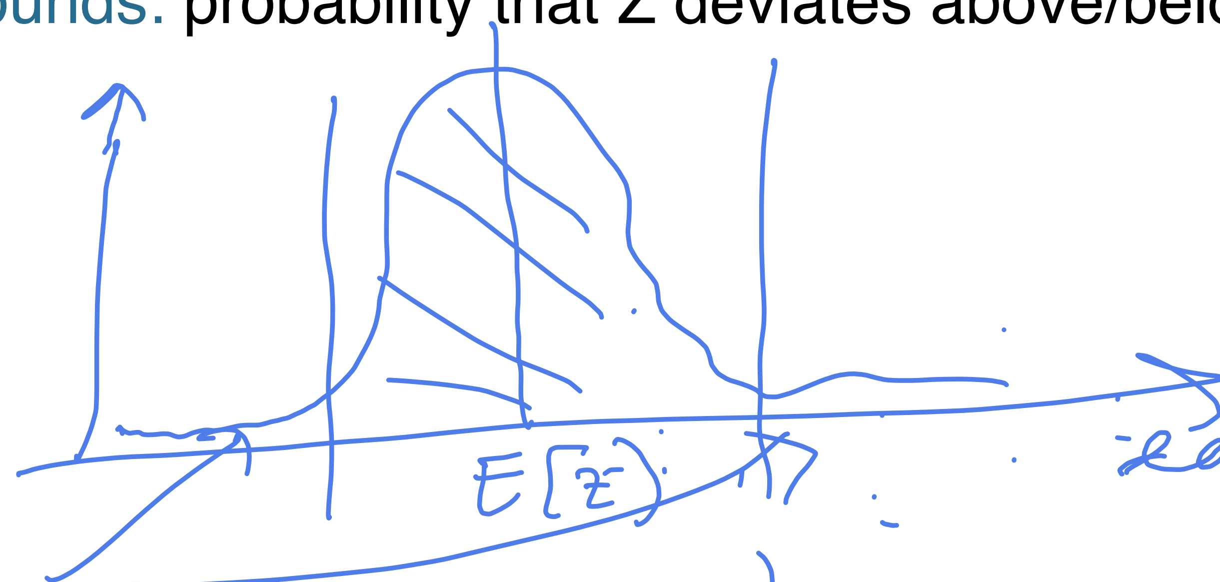
Chernoff bounds

$$z_i = \begin{cases} 1 & \text{prob } p_i \\ 0 & \text{prob } 1-p_i \end{cases}$$

- $Z_1, Z_2, \ldots, Z_n$ are independent random variables.
- Z_i is 1 with probability p_i, 0 otherwise
- $Z = Z_1 + Z_2 + \ldots + Z_n$ is a random variable, with $E[Z] = \sum_{i=1\ldots n} p_i$ = average value of random vars.

intuition: the fluctuation of the variables "cancel out", hence the value of Z is close to its expectation with high probability.

Chernoff bounds: probability that Z deviates above/below its expectation



each of z_i

tail

How likely is $z_1, z_2, \ldots z_n$ in the shaded part? ~ close to the expectation

Chernoff bounds

- $Z_1, Z_2, \ldots, Z_n$ are independent random variables.
- Z_i is 1 with probability p_i, 0 otherwise
- $Z = Z_1 + Z_2 + \ldots + Z_n$ is a random variable, with $E[Z] = \sum_{i=1\ldots n} p_i$

intuition: the fluctuation of the variables "cancel out", hence the value of Z is close to its expectation with high probability.

Chernoff bounds: probability that Z deviates above/below its expectation

Probability that Z is larger:

how likely is the value of Z to be a lot higher than E[Z]

Let $\mu \geq E[Z]$ for any $\delta > 0$

$$Pr\left(Z > (1+\delta)\mu\right) < \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^{\mu}$$

a "lot" example Z is more than 1.5 E[Z]

Z is smaller:

Let $\mu \leq E[Z]$ for any $1 > \delta > 0$

$$Pr\left(Z < (1-\delta)\mu\right) < e^{-\frac{1}{2}\mu\delta^2}$$

Load balancing

Analysis.

- Let X_i = number of jobs assigned to processor i.
- Let $Y_{ij} = 1$ if job j assigned to processor i, and 0 otherwise.
- We have $\mathrm{E}[Y_{ij}] = 1/\mathrm{n}$.
- Thus, $X_i = \sum_j Y_{ij}$, and $\mu = \mathrm{E}[X_i] = 1$.
- Applying Chernoff bounds with $\delta = \mathrm{c} - 1$ yields $Pr([X_i > c]) < \dfrac{e^{c-1}}{c^c}$

plug in the appropriate values to μ and
δ in the Chernoff bound

Load balancing

Analysis.

- Let X_i = number of jobs assigned to processor i.
- Let $Y_{ij} = 1$ if job j assigned to processor i, and 0 otherwise.
- We have $E[Y_{ij}] = 1/n$.
- Thus, $X_i = \sum_j Y_{ij}$, and $\mu = E[X_i] = 1$.
- Applying Chernoff bounds with $\delta = c - 1$ yields $Pr([X_i > c]) < \dfrac{e^{c-1}}{c^c}$
- let's figure out what is c:
- Let $\gamma(n)$ be number x such that $x^x = n$, and choose $c = e\,\gamma(n)$.

$$Pr\left[X_i > c\right] < \left(\frac{e^{c-1}}{c^c}\right) < \left(\frac{e}{c}\right)^c = \left(\frac{1}{\gamma(n)}\right)^{e\gamma(n)} < \left(\frac{1}{\gamma(n)}\right)^{2\gamma(n)} = \frac{1}{n^2}$$

- Union bound $\Rightarrow$ with probability $\geq 1 - 1/n$ no processor receives more than $e\gamma(n) = \Theta(\log n / \log\log n)$ jobs.

Load balancing: many jobs

Theorem. Suppose the number of jobs $m = 16\, n \ln n$. Then on average, each of the n processors handles $\mu = 16 \ln n$ jobs. With high probability, every processor will have between half and twice the average load.

Pf.

- Let X_i , Y_{ij} be as before.
- Applying Chernoff bounds with $\delta = 1$ yields

$$\Pr[X_i > 2\mu] \;<\; \left(\frac{e}{4}\right)^{16n \ln n} \;<\; \left(\frac{1}{e}\right)^{\ln n} \;=\; \frac{1}{n^2}$$

$$\Pr\left[X_i < \tfrac{1}{2}\mu\right] \;<\; e^{-\frac{1}{2}\,(\frac{1}{2})^2\, 16n \ln n} \;=\; \frac{1}{n^2}$$

- Union bound $\Rightarrow$ every processor has load between half and twice the average with probability $\geq 1 - 2/n$. ▪

Packet Routing

Communication network: s wants to send data to t, data is discretized into packets.

- directed graph G(V,E)
- node s sends packet to node t along specified path P
- at any point in time there may be many packets associated with different sources, destinations and paths

- an edge e can transmit *one packet at a time*
- e maintains a *queue* for packets waiting to traverse it

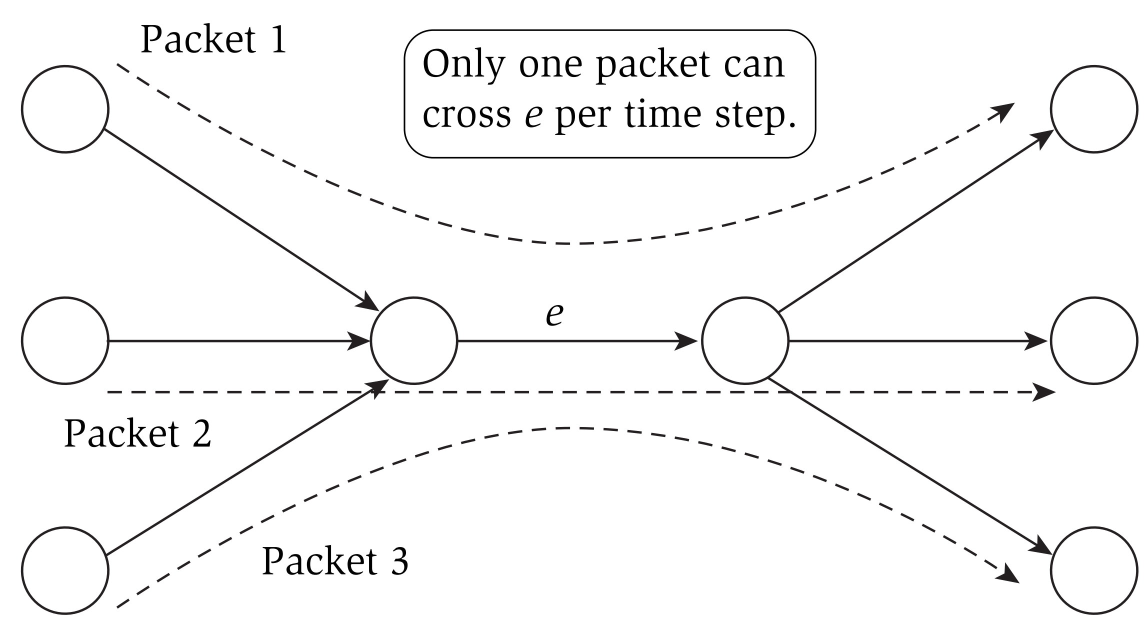


Figure 13.3 Three packets whose paths involve a shared edge e.

image from Kleinberg-Tardos ch. 13.11 page 763.

Packet scheduling

Goal: find the minimum number of time steps necessary to send all packets through the network

goal: clever routing decisions to minimize arrival times

- release time of packets at the source

→ delay the packet by some r timesteps

- queue management policy at each edge

→ multiple packets

→ which one should we send first.

Packet scheduling example

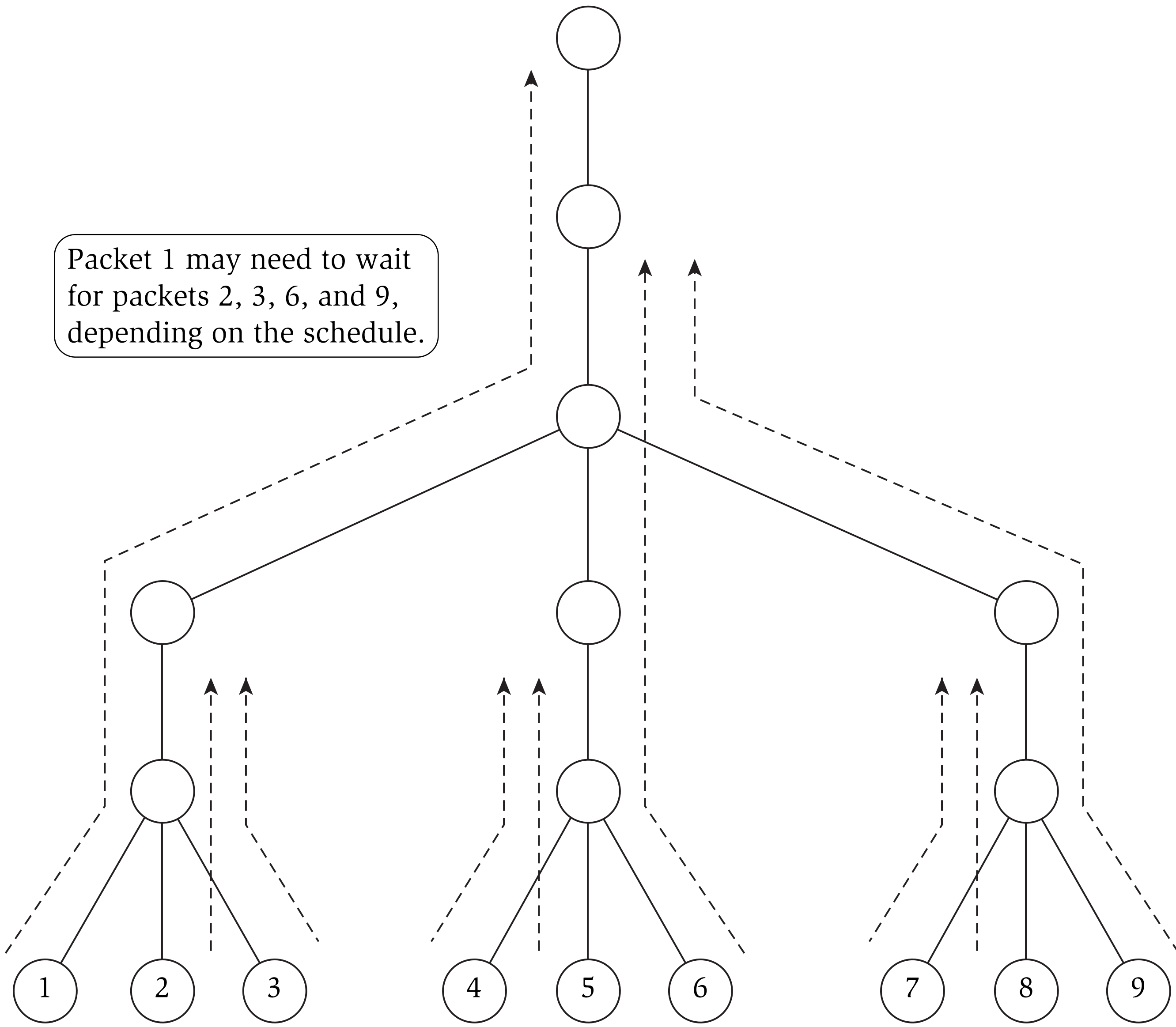


Figure 13.4 A case in which the scheduling of packets matters.

Each packet is traversing to the destination pointed by the arrow.

policy 1: send packet in queue that is *nearest* to its destination

- 9 steps for packet 1

policy 2: send packet in queue that is *furthest* from its destination

- packet 1 doesn't wait, 5 steps
- at most 6 steps for every packet

Generalization: tree of height h, nodes at every other level have k children. Then policy 1 yields $\Omega(hk)$ policy 2 yields $\Omega(h+k)$

image from Kleinberg-Tardos ch. 13.11 page 764.

Schedules and their duration

Given graph G, packets 1, 2,.., N and associated paths P_1, P_2, …, P_N

packet schedule: specifies for each edge e and time step t which packet will cross edge e at time t.

constraints:

- at most one packet can cross e at any time
- packet i will cross e if the edge is on path P_i
- i can pass e at time t, if the schedule has caused it to reach e prior to t

duration: the number of steps until every packet reaches its destination

→ time that the slowest packet arrives

problem: find the schedule with shortest duration

Schedules and their duration

Given graph G, packets 1, 2,.., N and associated paths P_1, P_2, …, P_N

packet schedule: specifies for each edge e and time step t which packet will cross edge e at time t.

duration: the number of steps until every packet reaches its destination

bounds on the duration:

- dilation d of paths P1, P2, …, PN is the max length of Pi
- congestion c maximum number of paths that have any single edge in common

observation: duration is at least max(d,c), thus $\Omega(d+c)$

Leighton, Maggs, Rao (1988): duration and congestion are in fact the only obstacles to a fast schedule.

they construct a schedule that is O(d+c)

- schedule is complicated, proof is difficult

Randomized packet routing

Naive random algorithm:

at each step t each edge e transmits a random packet in its queue

worst case: packet is delayed at every edge by c-1 others, and traverses a path of length d — O(dc)

intuition: timing of packet arrival at edges is poor. → packets arrive at the same time

We should delay by some random amount the release of packets at their source.

goal: make sure that packets don't arrive at the same time.

Randomized packet routing - delayed release

Delayed Release random algorithm:

input: r

- each packet i:
  - choose random delay time t_i between 1 and r (r is chosen in some clever way)
  - wait at source for t_i time steps
  - move one edge at a time until destination is reached

duration? if we are lucky and avoid collisions?

collision: only one packet at a time in the queue for each edge

$\leq r + d$

worst case start time

length of longest path

Randomized packet routing - delayed release

Delayed Release random algorithm:

input: r

- each packet i:
    - choose random delay time t_i between 1 and r (r is chosen in some clever way)
    - wait at source for t_i time steps
    - move one edge at a time until destination is reached

duration: at most r+d

- assumes that the random start times spread out the packets so that there is no collision.
- would need very large r to keep it collision free

Randomized packet routing - delayed block-release

Delayed Block-Release random algorithm:

input: parameter b , r

block: group intervals of b consecutive time steps into each block

- each packet i:
  - choose random delay time t_i between 1 and r
  - wait at source for t_i *blocks*
  - move forward one edge *per block* until destination is reached

duration?

- b(r+d) → duration for the slowest packet (if lucky with congestion) is worse than before
  → prob of avoiding congestion is higher

tradeoff between the size of b, r and prob of no collisions

Delayed block-release analysis

Let *E* be the event that more than b packets are at the same edge at the start of the same block.

- this implies that a packet would have to wait more than b steps

observation: if E does not occur, then the duration is O(b(r+d))

goal: find r and b such that Pr(E) is low, but b(r+d) is small.

Delayed block-release analysis – very high level overview

Let E be the event that more than b packets are at the same edge at the start of the same block.

↳ block size is b

⇒ within one block a queue ≤ b items is always resolved.

- this implies that a packet would have to wait more than b steps

observation: if E does not occur, then the duration is O(b(r+d))

goal: find r and b such that Pr(E) is low, but b(r+d) is small.

N_{et} = random variable, number of packets at edge e at time block t.

- we need to keep $Pr(N_{et} > b)$ low
- we can compute $E[N_{et}] \leq \frac{c}{r}$
- turns out it works if

$$3 = \frac{c}{q \log(mN)}$$

Leighton, Maggs, Rao: with high probability the expected duration is O(c+d log(mN))

Routing packets at the network layer in computer networks

Assume a package (i.e. information encoded in bits) is sent from host *s* to destination *t*.

- network consists of routers connected by physical links
- traversing each link has some cost (may be related to length, congestion, fee)
- goal is to send the package along a minimum cost route

Routing packets at the network layer in computer networks

Assume a package (i.e. information encoded in bits) is sent from host *s* to destination *t*.

- network consists of routers connected by physical links
- traversing each link has some cost (may be related to length, congestion, fee)
- goal is to send the package along a minimum cost route

- looks like an obvious application of Dijkstra's!
  - instead of finding shortest paths *from s to* each node (including *t*), we find shortest paths *from* each node *to t*.
  - how do you modify Dijkstra's to achieve this?

Routing packets – Link-state routing protocol

Assumes each router has *complete knowledge* of the network

- given a small (sub)network this is not an irrational assumption
- Each router *r* can run a shortest paths algorithm with itself as the source to find a path to *t*.

Changes in the network:

- Links change dynamically (e.g. goes down, gets congested)
- routers send periodical updates about their connecting links (neighbors) to every other router.

Protocol (in router *r*):

- when package arrives, compute shortest path to its destination *t*
- forward to the first link (next router) along this path
- periodically check state of neighbor links and broadcast to the other routers

Routing packets – Distance vector protocol

Assume routers have only knowledge about their *immediate* neighbors.

- *decentralized* approach
- makes sense for large networks or when link information is proprietary (often combined with something called border-gateway protocol)

Router *r* receives package *p* to be sent to destination *t*

- which neighbor to forward to?
- try to think of which one is the first edge along the shortest path from *r* to *t*

Routing packets – Distance vector protocol

Protocol (for router r):

- *r* maintains routing table with its currently known distance to each destination
  - for each neighbor *r'*, if the packet were to be routed through that neighbor the path length would be
    $$\pi(r) = d(r') + cost(r, r')$$
  - use the neighbor with least length
- when its routing table gets updated, *r* broadcasts new table to each neighbor
- when receiving an update from a neighbor, *r* updates its own table

- there are some pitfalls (e.g. there is a possibility for infinite loops) and mechanisms to avoid them.