L2 - Birthday Paradox and Coupon Collectors
[Jeff Phillips - Utah - Data Mining]

Universe of n elements [n]
 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
A "trial" draws a random element from [n].
After k trials, what phenomenon occur?

Birthday Paradox:
  after about k = sqrt{n} trials some element appears twice

Coupon Collectors:
  after about k = n log n trials, we see all elements
  each element appears on average log n times

Modeling:
[n] = set of all IP addresses
    = set of all words (or consecutive set of 3 words) in dictionary
    = set of all "types" of costumers
    = set of all products on Amazon
    = hash table buckets

    = birthdays of people in room (this room)
n = 365 (ignore leap year) assume each day equally likely

2 people
Pr[Alice + Bob have same birthday] == ? = 1/365
-->
Pr[Alice + Bob have different birthdays] =
  1-1/n = 1 - 1/365 ~= 0.997

k people
(k choose 2) = k(k-1)/2 ~= k^2/2 pairs of people
(independence) -->
Pr[no pair has same birthday] ~=~ (1-1/n)^{k choose 2}
                              ~= (1-1/n)^{k^2/2}
  ~= 0.997^{253} = 0.467
 (n = 365, k = 23)
Pr[some pair has same birthday] ~= 1-(1-1/n)^{k^2/2} ~= 0.532
   > 50%

<run class simulation>

* independence? (leap year, twins, more in spring?)
      Sometimes can force independence (or 2-way independence)
      when some collisions are more likely, these often govern probability, to a degree
      (1/4) + (3/4) {1/(n-1), 1/(1-n), ...}
         --> Prob 1/16^{k^2}

* sloppy -> k=n+1 --> (k=366, n=365) 1-(1-1/n)^{k choose 2} = 1-(0.997)^{66795} < 1
     very small, but < 1, so must be wrong.

   1 - ((n-1)/n)^{k-1} * ((n-2)/(n-1))^{k-2} * ...
 = 1 - prod_{i=1}^{k-1} ((n-i-1)/(n-i))^{k-i}

where the n-1 term is (n-(n-1)-1) / (n-(n-1)) = 0/1 = 0.

------->
  k = sqrt{2n}
  1 - (1 - 1/n)^{k choose 2} ~~ 1 - (1-1/n)^n ~~ 1 - 1/e ~~ 0.63

Not much deviation from
  happens 28% with between 18 and 28 people.
  happens 96% before 50 people

----

[n] = set of coupons in cereal box "collect them all!"
     = (all "types" of customers)

Pr[all coupons after k trials]
  if k < n --> 0
  too hard...
Pr[we see a new coupon | seen t]
  = (n-t)/n = p_t

Given seen t coupons, expected time to see new one
  T_t = 1/p_t

Expected time to all coupons:
  sum_{t=0}^{n-1} T_t
 =sum_{t=0}^{n-1} (n/(n-t))
 =n * sum_{t=1}^n (1/t)
 =n * H_n   the "nth Harmonic Number"

H_n = gamma + ln n + o(1/n)
      gamma ~~ 0.577 "Euler-Masheroni constant"

--> k = n * H_n ~ n(gamma + ln n)

<run class simulation, w/ months>

 * some events more/less likely.
   --> dominated by min-probability (p^* = min_i p_i) event
   k ~~ (1/p*) ln n

 * all "nice" events that occur with probability at least p
   k ~~ (1/p) log (1/p)

-------->
 * about n ln n trials to hit all events, not n. Extra log n factor.
 * all "nice" p-probability events with about ((1/p) log (1/p)) samples.