Graduate Algorithms

CS673-20016F-02

Probabilistic Analysis

David Galles

Department of Computer Science
University of San Francisco
02-0: Hiring Problem

- Need an office assistant
  - Employment Agency sends one candidate every day
  - Interview that person, either hire that person (and fire the old one), or keep old person
  - Always want the best person – always hire if interviewee is better than current person
HIRE-ASSISTANT(n)

best <- 0

for i <- 1 to n do
    if candidate[i] is better than candidate[best]
        best <- i
    hire candidate i

- Cost to interview candidate is $C_i$
- Cost to hire a candidate is $C_h$
- Assume $C_i$ is much less than $C_h$
- Total cost: $O(C_i \ast n + C_h \ast m)$, where $m = \#$ of hirings
02-2: Hiring Problem

- Best case cost?
- Worst case cost?
- Average cost?
02-3: Hiring Problem

- Best case cost? $C_i \times n + C_h$
- Worst case cost? $C_i \times n + C_h \times n$
- Average cost?
  - Assume applicants come in random order
  - Each permutation of applicants is equally likely
• Indicator variable associated with event $A$:

\[ I\{A\} = \begin{cases} 1 & \text{if } A \text{ occurs} \\ 0 & \text{if } A \text{ does not occur} \end{cases} \]

• Example: Flip a coin: $Y$ is a random variable representing the coin flip

\[ X_H = I\{Y = H\} = \begin{cases} 1 & \text{if } Y = H \\ 0 & \text{otherwise} \end{cases} \]
02-5: Probability Review

- Expected value $E[]$ of a random variable
  - Value you “expect” a random variable to have
  - Average (mean) value of the variable over many trials
  - Does not have to equal the value of any particular trial
    - Bus example(s)
Expected value $E[]$ of a random variable

$$E[X] = \sum_{\text{all values } x \text{ of } X} x \times Pr\{X = x\}$$

When we want the “average case” running time of an algorithm, we want the Expected Value of the running time
\[ X_H = I\{Y = H\} \]

\[
E[X_H] = E[I\{Y = H\}] \\
= 1 \times Pr\{Y = H\} + 0 \times Pr\{Y = T\} \\
= 1 \times \frac{1}{2} + 0 \times \frac{1}{2} \\
= \frac{1}{2}
\]
Expected # of heads in $n$ coin flips

- $X = \# \text{ of heads in } n \text{ flips}$
- $X_i = \text{indicator variable: coin flip } i \text{ is heads}$
02-9: Probability Review

- Expected # of heads in \( n \) coin flips
  - \( X = \) # of heads in \( n \) flips
  - \( X_i = \) indicator variable: coin flip \( i \) is heads

\[
E[X] = E \left[ \sum_{i=1}^{n} X_i \right] \\
= \sum_{i=1}^{n} E[X_i] \\
= \sum_{i=1}^{n} \frac{1}{2} = \frac{n}{2}
\]
For any event $A$, indicator variable $X_A = I\{A\}$

$$E[X_A] = Pr\{A\}$$

$$E[X_A] = 1 \times Pr\{A\} + 0 \times Pr\{\neg A\}$$

$$= Pr\{A\}$$
02-11: Hiring Problem

- Calculate the expected number of hirings
  - \( X = \# \text{ of candidates hired} \)
  - \( X_i = I\{\text{Candidate } i \text{ is hired}\} \)
  - \( X = X_1 + X_2 + \ldots + X_n \)

\[ E[X] = \]
02-12: Hiring Problem

- Calculate the expected number of hirings
  - \( X = \# \text{ of candidates hired} \)
  - \( X_i = I\{\text{Candidate } i \text{ is hired}\} \)
  - \( X = X_1 + X_2 + \ldots + X_n \)

\[
E[X] = E\left[ \sum_{i=1}^{n} X_i \right] = \sum_{i=1}^{n} E[X_i]
\]

- What is \( E[X_i] \)?
What is $E[X_i]$?

- $E[X_i] = \text{Probability that the } i\text{th candidate is hired}$
- When is the $i$th candidate hired?
What is $E[X_i]$?

- $E[X_i] = \text{Probability that the } i\text{th candidate is hired}$
- $i\text{th candidate hired when s/he is better than the } i-1 \text{ candidates that came before}$
- Assuming that all permutations of candidates are equally likely, what is the probability that the $i\text{th candidate is the best of the first } i \text{ candidates}$?
What is \( E[X_i] \)?

- \( E[X_i] \) = Probability that the \( i \)th candidate is hired
- \( i \)th candidate hired when s/he is better than the \( i - 1 \) candidates that came before
- Assuming that all permutations of candidates are equally likely, what is the probability that the \( i \)th candidate is the best of the first \( i \) candidates?
  - \( \frac{1}{i} \)
Probability that the $i$th candidate is best of first $i$ is $\frac{1}{i}$

- Sanity Check: (Doing a few concrete examples as a sanity check is often a good idea)
  - $i = 1$, probability that the first candidate is the best so far = $1/1 = 1$
  - $i = 2$: (1,2), (2,1) In one of the two permutations, 2nd candidate is the best so far
  - $i = 3$: (1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1) In two of the 6 permutations, the 3rd candidate is the best so far

- Note that a few concrete examples do not prove anything, but a counter-example can show that you have made a mistake
Now that we know that $E[X_i] = \frac{1}{i}$, we can find the expected number of hires:

$$
E[X] = E\left[ \sum_{i=1}^{n} X_i \right] = \sum_{i=1}^{n} E[x_i] = \sum_{i=1}^{n} \frac{1}{i} = \ln n + O(1) + O(1/\ln n) \in O(\log n)
$$
In average-case analysis, we often assume that all inputs are equally likely.

In actuality, some inputs might be much more likely:
- If we’re really unlucky, the most likely inputs can be the most costly (as in some implementations of quicksort).

What can we do?
In average-case analysis, we often assume that all inputs are equally likely.

In actuality, some inputs might be much more likely.
  - If we’re really unlucky, the most likely inputs can be the most costly (as in some implementations of quicksort).

What can we do?
  - Force all inputs to be equally likely, by randomizing the input.
In the hire-assistant problem, we can first randomly permute the lists of candidates, and then run the algorithm. Then, for any input, we’d be guaranteed that the expected number of hires would be $\ln n + O(1)$. How can we randomly permute a list, so that every permutation is equally as likely? That is, how can we shuffle a list, so that every permutation is equally likely? Assume that we have a good random number generator.
To create a random permutation (method 1):

- Assign each element in the list a random priority
- Sort based on the priority

\[
n \leftarrow \text{length}(A)
\]
\[
\text{for } i \leftarrow 1 \text{ to } n \text{ do}
\]
\[
\text{Priority}[i] = \text{Random}(1, n^3)
\]
\[
\text{sort } A \text{ (using Priority as keys)}
\]

- Why \( n^3 \)?
- Time?
To create a random permutation (method 2):

\[
\text{n} \leftarrow \text{length}(A)
\text{for } i \leftarrow 1 \text{ to } n \text{ do }
\text{swap}(A[i], A[\text{Random}(i,n)])
\]
02-23: On-line Hiring Problem

- Interview candidates one at a time
- After each person is interviewed:
  - Tell them at once they are not wanted
  - Hire them (and stop the interview process)
- How can we maximize the probability that we get the best person (assume that they come in random order – we can always randomize the input to insure this)
Algorithm:

- Interview first $k$ candidates, reject them all
- Continue to interview candidates, hiring the first one that is better than anyone seen so far

Problems? Can we do better?
02-25: On-line Hiring Problem

- Interview first $k$ candidates, reject them all
- Continue to interview candidates, hiring the first one that is better than anyone seen so far

Analysis:

- The bigger $k$ is, the larger the chance that we see the best person in the first $k$ (and don’t hire the best person).
- The smaller $k$ is, the larger the chance that we stop too soon.
- How should we pick $k$?
02-26: On-line Hiring Problem

- \( S \) = we pick the best applicant
- \( S_i \) = the best applicant is \( i \), and we pick \( i \).

\[
Pr\{S\} = \sum_{i=k+1}^{n} Pr\{S_i\}
\]

- Why \( k + 1 \) instead of 1?
- When is the best person picked?
02-27: **On-line Hiring Problem**

\[ \Pr\{s\} = \sum_{i=k+1}^{n} \Pr\{S_i\} \]

- Why \( k + 1 \) instead of 1?
  - \( \Pr\{S_i\} = 0 \) if \( i < k \), since we never pick the first \( k \) people

- When is the best person picked?
  - If the best person is interviewed, s/he will be picked. The best person is interviewed when candidates \( k+1..best-1 \) are all worse than the best in \( 1..k \)
On-line Hiring Problem

- $B_i$ == $i$th candidate is the best
- $O_i$ == none of applicants in $k + 1..i - 1$ are picked

$S_i$ (in terms of $B_i$ and $O_i$) = ?
02-29: On-line Hiring Problem

- \( B_i \) = \( i \)th candidate is the best
- \( O_i \) = none of applicants in \( k + 1 \ldots i - 1 \) are picked

\[ S_i = B_i \land O_i \]

\[
Pr\{S_i\} = Pr\{B_i \land O_i\} \\
= Pr\{B_i\} \ast Pr\{O_i|B_i\} \\
= Pr\{B_i\} \ast Pr\{O_i\} \\
= (1/n) \ast k/(i - 1)
\]
How do we find a value of a variable to maximize a function?
02-31: On-line Hiring Problem

Hard to take a derivative of a summation. However:

\[
\int_{m}^{n+1} f(x) \, dx \leq \sum_{i=m}^{n} f(i) \leq \int_{m-1}^{n} f(x) \, dx
\]

(if \( f(x) \) is monotonically decreasing)

Looking at just the lower bound:

\[
\frac{k}{n} \int_{k}^{n} \frac{1}{x} \, dx \leq \frac{k}{n} \sum_{i=k}^{n-1} \frac{1}{i}
\]

\[
\frac{k}{n} (\ln n - \ln k) \leq Pr\{S\}
\]
Maximizing the lower bound:

- To maximize $k/n(\ln n - \ln k)$: Take first derivative with respect to $k$, set to 0.
- (recall the product rule for derivatives: $D[f(k)g(k)] = D[f(k)]g(k) + f(k)D[g(k)]$)
To maximize $\frac{k}{n}(\ln n - \ln k)$: Take first derivative with respect to $k$, set to 0.

(recall the product rule for derivatives: 
$D[f(k)g(k)] = D[f(k)]g(k) + f(k)D[g(x)]$)

$$\frac{1}{n}(\ln n - \ln k - 1) = 0$$

$$\ln k = \ln n - 1$$

$$\ln k = \ln n - \ln e$$

$$\ln k = \ln \frac{n}{e}$$

$$k = \frac{n}{e}$$
On-line Hiring Problem

- Interview just under 1/3 of the applicants (hiring none of them)
- Hire the first person better than anyone seen so far
- Probability of getting the best person \( \geq \frac{n/e}{n(\ln n - \ln(n/e))} = \frac{1}{e(\ln e)} = \frac{1}{e} \approx 0.37 \)