Probabilistic Analysis

Department of Computer Science, Tsinghua University

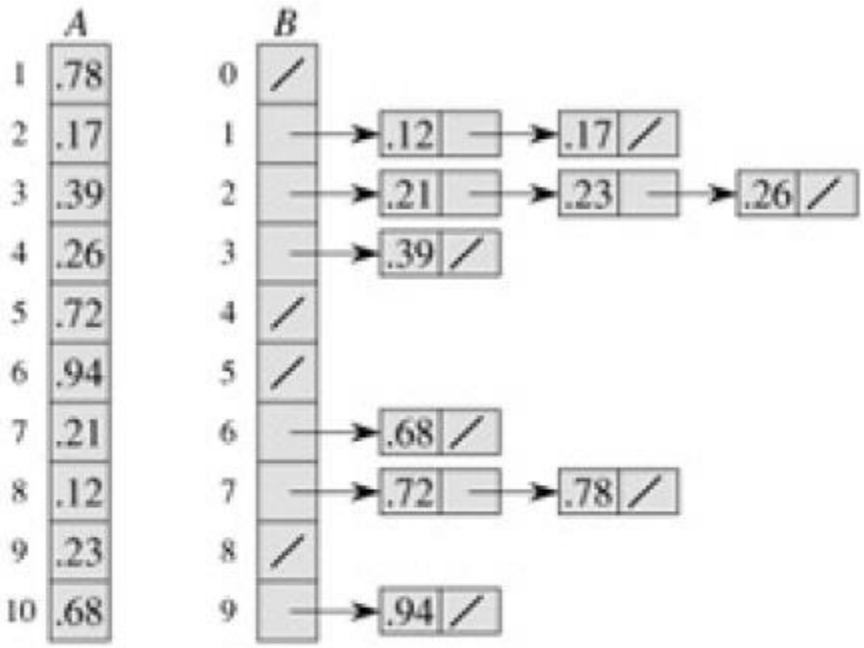
Review

- Probabilistic analysis is the use of probability in the analysis of problems.
- ***Running time:***
  - associated with counting numbers, i.e., the number of inversions in the input array, the number of elements checked etc.
  - ***Expected running time:*** over the distribution of the possible inputs (***average-case running time***).
- By probabilistic models:
  - Identify the Bernoulli trial: at the i^{th} stage, the Bernoulli trial is to hit a new bin with $i-1$ old bins and $\mathrm{n}-i+1$ new bins.
- By indicator random variables:
  - To count the # of inversions X, check each single pair X_{ij}
  - To count the # of checked elements X, consider each element X_i.

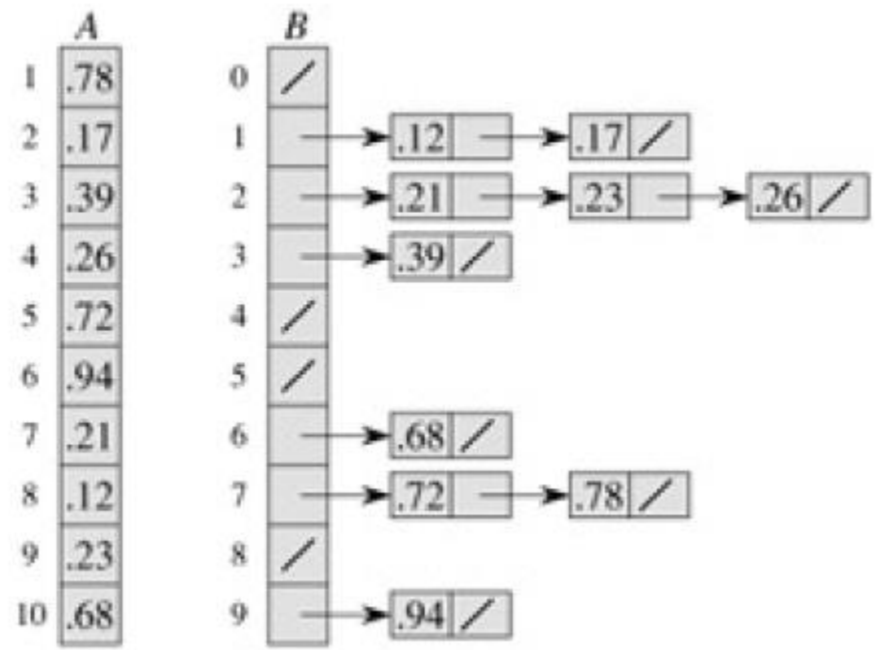
Bucket Sort

- ***Input***: $A[1\,..\,n]$, where $A[j] \in [0,1)$ and distributed uniformly.
- ***Output***: array $B[0\,..\,n-1]$ of sorted linked lists.

- ***Efficiency***: if $A[j] \in [0,1)$ and distributed uniformly, we can show that $\mathrm{E}[T(n)] = \theta(n)$.
- A linear time algorithm, **not** comparison-based.



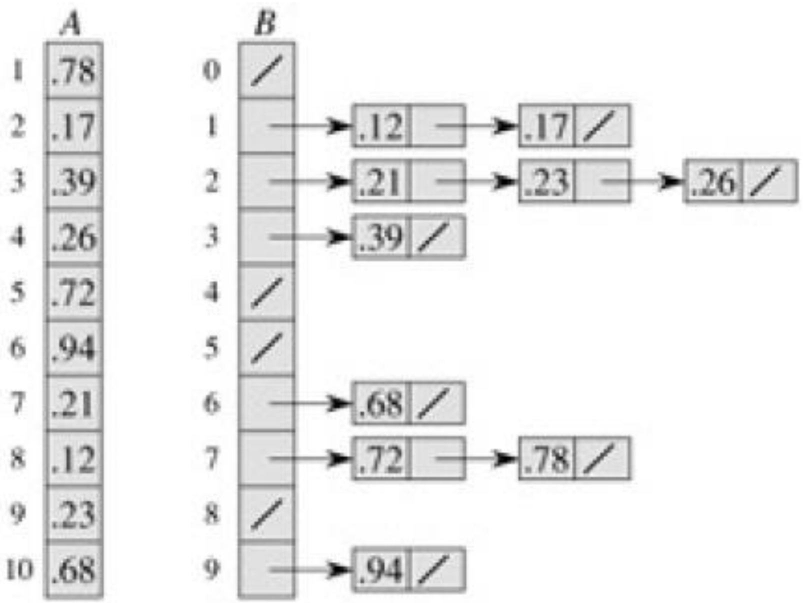
Bucket Sort




```
BUCKET-SORT(A)
n ← length[A]
for i ← 1 to n
    do insert A[i] into list B[⌊n A[i]⌋]
for i ← 0 to n − 1
    do sort list B[i] with insertion sort
concatenate the lists B[0], B[1], . . ., B[n − 1] together in order
```

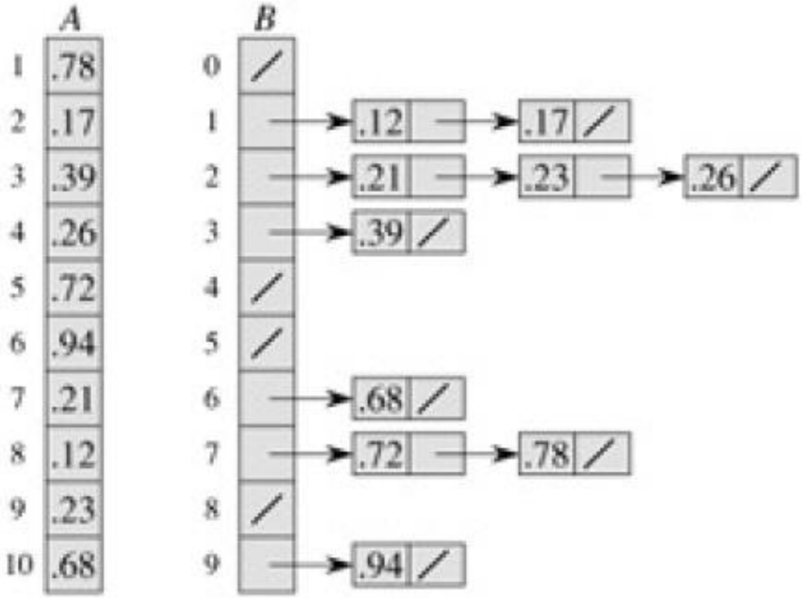
Bucket Sort

- ***Let*** n_i be the random variable denoting the number of elements placed in bucket $B[i]$, then $T(n) = \theta(n) + \sum_{i=0}^{n-1} O(n_i^2)$ and we need to solve $E[n_i^2]$.
- ***Step 1:*** transforming n_i^2 to n_i
  $E[n_i^2] = \text{Var}[n_i] + E^2[n_i]$
- ***Step 2:*** describing n_i using indicator random variables. (Hint: check each element?)



Bucket Sort

- ***Step 3:*** calculating $E[X_{ij}]$ and $\mathrm{Var}[X_{ij}]$



- ***Step 4:*** calculating $E[n_i]$

- ***Step 5:*** calculating $\mathrm{Var}[n_i]$

Hiring Problem

- **Problem:** Suppose that you need to hire a new office assistant. The employment agency sends you one candidate each day. After the interview, you must decide whether to hire him/her or not. You need to pay the agency for each candidate. You need to pay more, if you hire somebody.
- **Goal:** To have the best possible person for the job at all time.
- **Strategy:** assume that the candidates are numbered 1 to n. After interviewing candidate i, if i is the best one you have seen so far, hire i.
- **Cost:** estimate what the price will be?

Hiring Problem

```
HIRE-ASSISTANT(n)
best = 0  ▷ candidate 0 is a least-
  qualified dummy candidate
for i = 1 to n
    interview candidate i
    if i is better than best
        best = i
        hire candidate i
```

Hiring Problem

- Assume c_i is the cost for interview, c_h is the cost for hiring, then the total cost is $O(nc_i + mc_h)$, where m is the number of assistants hired during the process.
- *Worst-case analysis:* $O(nc_h)$
- *How to calculate the averaged-case?*
  - Input: a sequence of applicants
  - Let $\text{rank}(i)$ to denote the rank of applicant i among all applicants, then the input $< \text{rank}(1), \cdots, \text{rank}(n) >$ actually determines the number of hired applications.
  - In-class exercise: given a list of ranks, find the number of hired applicants.

Hire problem

- *How to calculate the averaged-case?*
  - The list of ranks $<\mathrm{rank}(1), \cdots, \mathrm{rank}(n)>$ is a permutation of the list $< 1, \cdots, n >$.
  - A_1 =<1,2,3,4,5,6,7,8,9,10>;   10 hires
  - A_2 =<10,9,8,7,6,5,4,3,2,1>;   1 hire
  - A_3 =<5,2,1,8,4,7,10,9,3,6>;   3 hires
  - We assume the applicants come in a random order, which means this list is equally likely to be any one of the $n!$ permutations.

- Given this distribution of inputs, we calculate the expectation of the cost with the help of indicator random variables.

Analysis (1)

- Given an input $< \mathrm{rank}(1), \cdots, \mathrm{rank}(n) >$, let X be the number of hired applicants.
  - $E[X] = \sum_{x=1}^{n} xPr\{X = x\}$, i.e., $Pr\{X = 1\} = \frac{(n-1)!}{n!} = \frac{1}{n}$
- Indicator random variable:
  - Hint: Check each applicant hired or not.

- Let X_i be the indicator random variable associated with the event in which candidate i is hired.
  - $X_i = I\{\text{candidate } i \text{ is hired}\} = \begin{cases} 1 & \text{if candidate } i \text{ is hired} \\ 0 & \text{if candidate } i \text{ is not hired} \end{cases}$
  - $X = X_1 + X_2 + \cdots + X_n$

Analysis (2)

- $E[X_i] = Pr\{\text{candidate } i \text{ is hired}\}$
- The first i candidates have appeared in a random order, candidate i has a probability of $1/i$ of being the best, so $E[X_i] = 1/i$.

- $E[X] = E\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} 1/i = \ln n + O(1)$ (harmonic series see A.7)

- Lemma 5.2: Assuming that the candidates are presented in a random order, algorithm HIRE-ASSISTANT has a total hiring cost of $O(c_h \ln n)$

On-line Hiring Problem

- **On-line version rules**:
  - Sometimes, the cost of hiring is too expensive (i.e., buy a house, marriage, etc.)
  - Only hire once.
  - After each interview we must either offer the position to the applicant or reject the applicant.

- **Simple strategy**:
  - Choose a positive integer $k < n$, interviewing and rejecting the first k applicants, and hire the first applicant who has a higher score than the first k applicants.

On-line Hiring Problem

- **Simple strategy**:
  - Choose a positive integer $k < n$, interviewing and rejecting the first k applicants, and hire the first applicant who has a higher score than the first k applicants.
- Does this strategy work?
  - It's possible that the best-qualified applicant is in the first k applicants.
  - It's possible the an applicant is hired before the best-qualified applicant.
  - k is an important parameter to this simple strategy.
  - This is also called ***the optimal stopping theory***.

On-line Hiring Problem

- Does this strategy work?
  - Chose $k = n/e$, the probability of hiring the best one is at least $1/e \approx 0.37$

- Analysis
  - S: the best-qualified applicant is hired.
  - S_i: the best-qualified applicant is the i^{th} applicant & hired.
  - B_i: the best-qualified applicant is the i^{th} applicant.
  - O_i: the best-qualified applicant is hired.
  - B_i and O_i are independent.