Linear Sorting Techniques

CLRS Reading: Sections 8.2, 8.3, 8.4, pages 194 ~ 212

Sorting with Special Knowledge

Counting-Sort(A, B, k)

- let C[0..k] be a new array
  - for i = 0 to k
    - C[i] = 0
  - for j = 1 to A.length
    - C[A[j]] = C[A[j]] + 1
    - // C[i] now contains the number of elements equal to i.
  - for i = 1 to k
    - C[i] = C[i] + C[i-1]
    - // C[i] now contains the number of elements less than or equal to i.
  - for j = A.length downto 1
    - B[C[A[j]]] = A[j]
    - C[A[j]] = C[A[j]] - 1
**Worst-Case Running Time of Counting-Sort**

- Let \( n \) be the number of entries in \( A \), each an integer in the range 0 to \( k \) for some integer \( k \).

```plaintext
Counting-Sort(A,B,k)
let C[0..k] be a new array
for i = 0 to k
  C[i] = 0
for j = 1 to A.length
  C[A[j]] = C[A[j]] + 1
// C[i] now contains the number of elements equal to i.
for i = 1 to k
  C[i] = C[i] + C[i-1]
// C[i] now contains the number of elements less than or equal to i.
for j = A.length downto 1
  B[C[A[j]]] = A[j]
  C[A[j]] = C[A[j]] - 1
```

**Punch Card Sort**

- There are two possibilities when sorting by column: most to least significant; or least to most significant digit. Suggestions?
**Seeing is Believing**

Radix-Sort($A, d$)

\[ \text{for } i = 1 \text{ to } d \]

use a stable sort
to sort array $A$
on digit $i$

<table>
<thead>
<tr>
<th>input array</th>
<th>first digit</th>
<th>second digit</th>
<th>third digit</th>
</tr>
</thead>
<tbody>
<tr>
<td>672</td>
<td>839</td>
<td>291</td>
<td>839</td>
</tr>
<tr>
<td>291</td>
<td>306</td>
<td>672</td>
<td>672</td>
</tr>
<tr>
<td>306</td>
<td>244</td>
<td>244</td>
<td>306</td>
</tr>
<tr>
<td>244</td>
<td>672</td>
<td>839</td>
<td>291</td>
</tr>
<tr>
<td>839</td>
<td>291</td>
<td>021</td>
<td>244</td>
</tr>
<tr>
<td>021</td>
<td>021</td>
<td>306</td>
<td>021</td>
</tr>
</tbody>
</table>

**Worst-Case Running Time of Radix-Sort**

*Lemma 8.3.*

Given $n$ $d$-digit numbers in which each digit can take on up to $k$ possible values, RADIX-SORT correctly sorts in $\Theta(d^*(n+k))$ time.

Radix-Sort($A, d$)

\[ \text{for } i = 1 \text{ to } d \]

use a stable sort
to sort array $A$
on digit $i$
Bucket Sort*

Bucket-Sort(A)
let B[0..n-1] be a new array
n = A.length
for i = 0 to n-1
    make B[i] an empty list
for i = 1 to n
    insert A[i] into list B[⌊n*A[i]⌋]
for i = 0 to n-1
    sort list B[i] with insertion sort
concatenate the lists B[0], B[1], ..., B[n-1]
Bucket Sort*

Bucket-Sort(A)
let B[0..n-1] be a new array
n = A.length
for i = 0 to n-1
make B[i] an empty list
for i = 1 to n
insert A[i] into list B[n*A[i]]
for i = 0 to n-1
sort list B[i] with insertion sort
concatenate the lists B[0], B[1], ..., B[n-1]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.38</td>
</tr>
<tr>
<td>2</td>
<td>.32</td>
</tr>
<tr>
<td>3</td>
<td>.63</td>
</tr>
<tr>
<td>4</td>
<td>.07</td>
</tr>
<tr>
<td>5</td>
<td>.30</td>
</tr>
<tr>
<td>6</td>
<td>.49</td>
</tr>
</tbody>
</table>

J.9

Bucket Sort*

Bucket-Sort(A)
let B[0..n-1] be a new array
n = A.length
for i = 0 to n-1
make B[i] an empty list
for i = 1 to n
insert A[i] into list B[n*A[i]]
for i = 0 to n-1
sort list B[i] with insertion sort
concatenate the lists B[0], B[1], ..., B[n-1]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.38</td>
</tr>
<tr>
<td>2</td>
<td>.32</td>
</tr>
<tr>
<td>3</td>
<td>.63</td>
</tr>
<tr>
<td>4</td>
<td>.07</td>
</tr>
<tr>
<td>5</td>
<td>.30</td>
</tr>
<tr>
<td>6</td>
<td>.49</td>
</tr>
</tbody>
</table>

J.10
Bucket Sort*

Bucket-Sort(A)

let $B[0..n-1]$ be a new array

$n = A.length$

for $i = 0$ to $n-1$

make $B[i]$ an empty list

for $i = 1$ to $n$

insert $A[i]$ into list $B[\lceil n\cdot A[i] \rceil]$

for $i = 0$ to $n-1$

sort list $B[i]$ with insertion sort

concatenate the lists $B[0], B[1], ..., B[n-1]$

\[
\begin{array}{c|c|c}
A & B \\
--- & --- \\
1 & .38 & 0 \rightarrow .38 \\
2 & .32 & 1 \rightarrow .32 \\
3 & .63 & 2 \rightarrow .63 \\
4 & .07 & 3 \rightarrow .07 \\
5 & .30 & 4 \\
6 & .49 & 5 \\
\end{array}
\]
Bucket Sort*

Bucket-Sort(A)
let B[0..n-1] be a new array
n = A.length
for i = 0 to n-1
make B[i] an empty list
for i = 1 to n
insert A[i] into list B[⌊n*A[i]⌋]
for i = 0 to n-1
sort list B[i] with insertion sort
concatenate the lists B[0], B[1], ..., B[n-1]

*What might happen in the worst case?
Bucket Sort*

Bucket-Sort(A)
let B[0..n-1] be a new array
n = A.length
for i = 0 to n-1
    make B[i] an empty list
for i = 1 to n
    insert A[i] into list B[ln*A[i]]
for i = 0 to n-1
    sort list B[i] with insertion sort
concatenate the lists B[0], B[1], ..., B[n-1]

Average Case Analysis of Bucket-Sort

\[
\Pi(n) = \Theta(n) + \sum_{0 \leq i \leq n-1} O(n^2)
\]

Cost of insertion sort on \(i^{th}\) bucket given bucket contains \(n_i\) elements
Cost of inserting into buckets and concatenating results
Summed over all the buckets
Average Case Analysis of Bucket-Sort

\[ E[\mathcal{T}(n)] = E[\Theta(n) + \sum_{0 \leq i \leq n-1} \Theta(n_i^2)] \]
\[ = E[\Theta(n)] + \sum_{0 \leq i \leq n-1} E[\Theta(n_i^2)] \]
\[ = \Theta(n) + \sum_{0 \leq i \leq n-1} \Theta(E[n_i^2]) \]
\[ = \Theta(n) + \sum_{0 \leq i \leq n-1} \Theta(2 - 1/n)* \]
\[ = \Theta(n) + n \Theta(2 - 1/n) \]
\[ = ? \]

*We are using the fact, established in the following slides, that \(E[n_i^2] = 2 - 1/n\).

\[ n_i = \sum_{1 \leq j \leq n} X_{ij} \]

where \(X_{ij}\) is the indicator function

\[ X_{ij} = \begin{cases} 1 & \text{if } A[j] \text{ falls into bucket } i \\ 0 & \text{otherwise} \end{cases} \]

Some observations concerning \(X_i\): \(E[X_i] = 1/n\); \(X_i^2 = 1/n\); and \(E[X_iX_j] = E[X_i]E[X_j] = 1/n^2\).
Since \( n_i = \sum_{1 \leq j \leq n} X_{ij} \)

\[
E[n^2] = E[(\sum_{1 \leq j \leq n} X_{ij})^2] \\
= E[\sum_{1 \leq j \leq n} \sum_{1 \leq k \leq n} X_{ij} X_{ik}] \\
= E[\sum_{1 \leq j \leq n} X_{ij}^2 + \sum_{1 \leq j \leq n} \sum_{1 \leq k \leq n, k \neq j} X_{ij} X_{ik}] \\
= E[\sum_{1 \leq j \leq n} X_{ij}^2] + \sum_{1 \leq j \leq n} \sum_{1 \leq k \leq n, k \neq j} E[X_{ij} X_{ik}] \\
= \sum_{1 \leq j \leq n} 1/n + \sum_{1 \leq j \leq n} \sum_{1 \leq k \leq n, k \neq j} 1/n^2 \\
= n (1/n) + n(n-1) 1/n^2 \\
= 1 + (n-1)/n \\
= 2 - 1/n
\]