More on Sorting and Maxima

CSCI2100 Tutorial 5
Hou Pong CHAN, Ken
Counting Sort

• Sort a set of integers in some range $[1, U]$

Initialize array $B$

| 7 | 2 | 6 | 4 | 8 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Scan through $A$

| 7 | 2 | 6 | 4 | 8 | 9 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |

Scan through $B$

| 2 | 4 | 6 | 7 | 8 | 9 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
Counting Sort

• Modify the counting sort to sort a multi-set of arbitrary objects based on their integer keys
• Multi-set: A set which allows duplicate elements
  • e.g., {1,1,2,3,5,8}
Sorting Problem (Small Domain) on a Multi-Set

• Problem Input:
  • A multi-set $S$ of $n$ integers (each in the range $[1, U]$) is given in an array of length $n$
  • The values of $n$ and $U$ are inside two registers

• Goal:
  • Arrange the elements of $S$ in non-decreasing order
Example

• $B$ acts as counters instead of flags

Initialize array $B$

<table>
<thead>
<tr>
<th>7</th>
<th>2</th>
<th>6</th>
<th>2</th>
<th>9</th>
<th>7</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
</table>

Scan through $A$

<table>
<thead>
<tr>
<th>7</th>
<th>2</th>
<th>6</th>
<th>2</th>
<th>9</th>
<th>7</th>
<th>1</th>
<th>2</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
</table>

Scan through $B$

| 1 | 2 | 2 | 2 | 6 | 7 | 7 | 9 | 1 | 3 | 0 | 0 | 0 | 1 | 2 | 0 | 1 |
Sorting-by-Key Problem (Small Domain)

• Problem Input:
  • A multi-set $S$ of $n$ 2-tuples (of integers) in an array of length $2n$
  • Each 2-tuples as a key-value pair, where the 1st position gives the key, 2nd position gives the value
  • All keys are in the range $[1, U]$
  • The values of $n$ and $U$ are inside two registers

• Goal:
  • Arrange the elements of $S$ in non-decreasing order by key
Example

• Consider a multi-set $S$
  $S = \{(9,1), (7,2), (2,4), (6,5), (2,6), (7,7), (1,8), (2,9)\}$

• Initially we will have the following array

<table>
<thead>
<tr>
<th>$k_1$</th>
<th>$v_1$</th>
<th>$k_2$</th>
<th>$v_2$</th>
<th>$k_3$</th>
<th>$v_3$</th>
<th>$k_4$</th>
<th>$v_4$</th>
<th>$k_5$</th>
<th>$v_5$</th>
<th>$k_6$</th>
<th>$v_6$</th>
<th>$k_7$</th>
<th>$v_7$</th>
<th>$k_8$</th>
<th>$v_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>1</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>1</td>
<td>8</td>
<td>2</td>
<td>9</td>
</tr>
</tbody>
</table>

• Rearrange the elements so that the keys are sorted:

<table>
<thead>
<tr>
<th>$k_1$</th>
<th>$v_1$</th>
<th>$k_2$</th>
<th>$v_2$</th>
<th>$k_3$</th>
<th>$v_3$</th>
<th>$k_4$</th>
<th>$v_4$</th>
<th>$k_5$</th>
<th>$v_5$</th>
<th>$k_6$</th>
<th>$v_6$</th>
<th>$k_7$</th>
<th>$v_7$</th>
<th>$k_8$</th>
<th>$v_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>2</td>
<td>9</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>2</td>
<td>9</td>
<td>1</td>
</tr>
</tbody>
</table>
Example

• Simply change $B$ to be counters?

Scan through $A$

<table>
<thead>
<tr>
<th>7</th>
<th>2</th>
<th>6</th>
<th>2</th>
<th>9</th>
<th>7</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
</table>

1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

• Not enough!
• Why? Because we need to track information regarding the values
Example

Compute $B$

\[
\begin{array}{cccccccccc}
9 & 1 & 7 & 2 & 2 & 4 & 6 & 5 & 2 & 6 & 7 & 7 & 1 & 8 & 2 & 9
\end{array}
\]

Compute the cumulative sum of $B$

\[
\begin{array}{cccccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9
\end{array}
\]

The keys in a sorted order

\[
\begin{array}{cccccccc}
1 & 2 & 4 & 5 & 7 & 8
\end{array}
\]

The cumulative sum encodes the last index a particular key will be found in the final sorted array
Example

Build up a new array $A'$ by repeating the following: for a key-value pair $(k, v)$ in $A$, move it to the $B'[k]$-th position in $A'$

Decrement the value in $B'$ to ensure that it always point to a valid, empty position in $A'$
Example

The second iteration

\[
\begin{array}{cccccccccccc}
9 & 1 & \textcolor{blue}{7} & 2 & 2 & 4 & 6 & 5 & 2 & 6 & 7 & 7 & 1 & 8 & 2 & 9 \\
1 & 4 & 4 & 4 & 4 & 5 & \textcolor{red}{7} & 7 & 7
\end{array}
\]

\[
\begin{array}{cccccccccccc}
A \\
A'
\end{array}
\]
Example

The third iteration

\[\begin{array}{ccccccccccc}
9 & 1 & 7 & 2 & 2 & 4 & 6 & 5 & 2 & 6 & 7 & 7 & 1 & 8 & 2 & 9 \\
\end{array}\]

\[\begin{array}{ccccccccccc}
1 & 4 & 4 & 4 & 4 & 4 & 5 & 6 & 6 & 7 & 7 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\end{array}\]

\[\begin{array}{ccccccccccc}
\text{A} \\
\text{A'} \\
\text{B'} \\
\end{array}\]
Example

The fourth iteration

```
<table>
<thead>
<tr>
<th>9</th>
<th>1</th>
<th>7</th>
<th>2</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>5</th>
<th>2</th>
<th>6</th>
<th>7</th>
<th>7</th>
<th>1</th>
<th>8</th>
<th>2</th>
<th>9</th>
</tr>
</thead>
</table>
A
```

```
| 1 | 3 | 4 | 4 | 4 | 4 | 5 | 6 | 7 | 7 |
B'
```

```
| 9 | 1 | 7 | 2 | 2 | 4 | 6 | 5 | 2 | 6 | 7 | 7 | 1 | 8 | 2 | 9 |
A

```

```
| 2 | 4 | 6 | 5 | 7 | 2 | 9 | 1 |
A'
```

```
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
```

```
| 1 | 3 | 4 | 4 | 4 | 6 | 7 | 7 |
B'
```

```
| 2 | 4 | 6 | 5 | 7 | 2 | 9 | 1 |
A'
```
Example

The fifth iteration
Example

The sixth iteration

The seventh iteration
Example

The eighth iteration
Time Complexity

• Step 1: scanning through $A$ to compute $B$ takes $O(n)$ time
• Step 2: computing the cumulative sum $B'$ takes $O(U)$ time
• Step 3: scanning $A$ and using $B'$ to copy elements over into $A'$ takes $O(n)$ time
• Overall time complexity: $O(n + U)$
Sorting a Multi-Set

• Regular Exercise 5 Problem 2
• Let $A$ be an array of $n$ integers
• Some of the integers may be identical
• Design a comparison-based algorithm to arrange these integers in non-descending order
• For example,
• $A = [35, 12, 28, 12, 35]$
• Output $[12, 12, 28, 35, 35]$
Solution

• Construct a new set $S$ of $n$ elements as follows:
  • The $i$-th element of $S$ equals $(i, v)$ where $v = A[i]$
• Create an array $B$ of length $n$ where $B[i]$ equals to the $i$-th element in $S$
• For example,
  • $A = [35,12,28,12,35]$
  • $S = \{(1,35), (2,12), (3,28), (4,12), (5,35)\}$
  • $B = [(1,35), (2,12), (3,28), (4,12), (5,35)]$
• Takes $O(n)$ time to generate $B$
Solution

- Apply merge sort to sort $B$
- Treat merge sort as a black box
- Replace the comparator of the merge sort

Diagram:
- Input: $B$
- Process: Merge Sort
- Output: Sorted $B$

Comparator:
- Input: $e_1, e_2$
- Process: The order between $e_1, e_2$
- Output: Our Comparator
Solution

• Our comparator compare two elements $e_1 = (i_1, v_1)$ and $e_2 = (i_2, v_2)$ as follows
  
  • If $v_1 < v_2$, then rule $e_1 < e_2$
  
  • If $v_1 > v_2$, then rule $e_1 > e_2$
  
  • If $v_1 = v_2$:
      
      • If $i_1 < i_2$, then rule $e_1 < e_2$
      
      • Otherwise, rule $e_1 > e_2$
Solution

• After merge sort

• $B = [(2,12), (4,12), (3,28), (1,35), (5,35)]$

• Easily generate the output array from $B$ in $O(n)$ time

• $[12,12,28,35,35]$
Time Complexity

• Generate B takes $O(n)$
• Merge sort takes $O(n \log n)$
• Generate the output from B takes $O(n)$
• Overall time complexity: $O(n \log n)$
Maxima

• Regular Exercise 5 Problem 5
• A point \((x, y)\) dominates another point \((x', y')\) if \(x > x'\) and \(y > y'\)

B dominates A, C dominates B, C dominates A
Maxima

- Let $S$ be a set of $n$ points
- A point $p \in S$ is a maximal point of $S$ if no point in $S$ dominates $p$

B and C are two maximal points
Maxima

• Design an algorithm to find all the maximal points of $S$ in $O(n \log n)$ time

• Solution:
  • Sort all the points by $x$-coordinate
  • Scan the points from the right (the largest $x$-coordinate)
Maxima

• Begin from the rightmost point $p_1 = (x_1, y_1)$
• Report $p_1$ as the maximal point
• Set $y_{max} = y_1$
Maxima

• The next point $p_2 = (x_2, y_2)$
• $y_2 < y_{\text{max}}$
• Ignore $p_2$
Maxima

• The next point \( p_3 = (x_3, y_3) \)
• \( y_3 > y_{\text{max}} \)
• Report \( p_3 \) as the maximal point
• Set \( y_{\text{max}} = y_3 \)
Maxima

• The next point $p_4 = (x_4, y_4)$
• $y_4 < y_{max}$
• Ignore $p_4$
Maxima

- The next point $p_5 = (x_5, y_5)$
- $y_5 > y_{max}$
- Report $p_5$ as the maximal point
- Set $y_{max} = y_5$
Maxima

• After scanning all the points
• Obtain all the maximal points as shown in orange
Time Complexity

• Sorting the points in $S$ takes $O(n \log n)$
• Scan all the points from the right takes $O(n)$
• Overall complexity: $O(n \log n)$