Lecture 10
Range minimum queries

Benjamin Sach

Preprocess an integer array $A$ (length $n$) to answer range minimum queries.

After preprocessing, a range minimum query is given by $\text{RMQ}(i, j)$, the output is the location of the smallest element in $A[i, j]$.

e.g.
- $\text{RMQ}(3, 7) = 6$, which is the location of the smallest element in $A[3, 7]$
- $\text{RMQ}(5, 11) = 8$, which is the location of the smallest element in $A[5, 11]$

We will discuss several algorithms which give trade-offs between space used, prep. time and query time.

Ideally we would like $O(n)$ space, $O(n)$ prep. time and $O(1)$ query time.

**Block decomposition**

$A_k$ is an array of length $\frac{n}{k}$ so that for all $i$, $A_k[i] = (x, v)$ where $v$ is the minimum in $A[ik, (i+1)k]$ and $x$ is its location in $A$.

We store $A_k$ for all $k = 1, 2, 4, 8 \ldots \leq n$.

How much space is this? $O(n)$ in total.

How quickly can we build them? $O(n)$ preprocessing time.

We construct the $A_k$ arrays bottom-up from these in $O(1)$ time.

How do we find $\text{RMQ}(i, j)$?

Find the largest block which is completely contained within the query interval but doesn’t overlap a block you chose before.

The minimum is the smallest in all these blocks because they cover the query.

Repeat:

How many blocks do we pick?

at most $2$ blocks of each size.

How do we find $\text{RMQ}(i, j)$?

Find the largest block which is completely contained within the query interval but doesn’t overlap a block you chose before.

The minimum is the smallest in all these blocks because they cover the query.

Repeat:

How many blocks do we pick?

at most $2$ blocks of each size.
**Block decomposition**

How do we find RMQ(1,9)?

**Repeat:** Find the largest block which is completely contained within the query interval but doesn’t overlap a block you chose before.

The minimum is the smallest in all these blocks because they cover the query.

<table>
<thead>
<tr>
<th>A16</th>
<th>A8</th>
<th>A4</th>
<th>A2</th>
<th>A</th>
<th>10,000 foot view</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://example.com/block_diagram.png" alt="Image" /></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

How many blocks do we pick?
- at most 2 blocks of each size
- so we have \(O(n)\) space, \(O(n)\) prep time, \(O(\log n)\) query time

---

**More space, faster queries**

**Key Idea** precompute the answers for every interval of length 2, 4, 8, 16…

<table>
<thead>
<tr>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://example.com/more_space_faster_queries.png" alt="Diagram" /></td>
</tr>
</tbody>
</table>

The array \(R_2\) stores RMQ\((i, i + 1)\) for all \(i\)
- \(R_4\) stores RMQ\((i, i + 3)\) for all \(i\)
- \(R_8\) stores RMQ\((i, i + 7)\) for all \(i\)
- \(R_{16}\) stores RMQ\((i, i + 15)\) for all \(i\)

We build \(R_{2k}\) for \(k = 2, 4, 8, 16 \ldots \leq n\) in \(O(n)\) time.

- each of the \(O(\log n)\) arrays uses \(O(n)\) space
- so \(O(n \log n)\) total space

---

**More space, faster queries**

**Key Idea** precompute the answers for every interval of length 2, 4, 8, 16…

<table>
<thead>
<tr>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://example.com/more_space_faster_queries.png" alt="Diagram" /></td>
</tr>
</tbody>
</table>

The array \(R_{2k}\) stores RMQ\((i, j)\) for all \(i, j\)
- \(R_k\) stores RMQ\((i, i + k - 1)\) for all \(i\)
- \(R_{2k}\) stores RMQ\((i, i + 2k - 1)\) for all \(i\)
- \(R_{4k}\) stores RMQ\((i, i + 4k - 1)\) for all \(i\)

We build \(R_{2k}\) for \(k = 2, 4, 8, 16 \ldots \leq n\) in \(O(n)\) time.

- each of the \(O(\log n)\) arrays uses \(O(n)\) space
- so \(O(n \log n)\) total space

---

**More space, faster queries**

**Key Idea** precompute the answers for every interval of length 2, 4, 8, 16…

<table>
<thead>
<tr>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://example.com/more_space_faster_queries.png" alt="Diagram" /></td>
</tr>
</tbody>
</table>

How do we compute RMQ\((i, j)\)?

If the interval length, \(\ell = (j - i + 1)\), is a power-of-two - just look up the answer these queries take \(O(1)\) time.

Otherwise, find the \(k = 2, 4, 8, 16 \ldots \) such that \(k \leq \ell < 2k\)

Compute the minimum of RMQ\((i, i + k - 1)\) and RMQ\((j - k + 1, j)\)

(These two queries take \(O(1)\) time)

This takes \(O(1)\) time but why does it work?
**Range minimum query (intermediate) summary**

Preprocess an integer array $A$ (length $n$) to answer range minimum queries...

$A = [5, 7, 8, 2, 32, 10, 85, 5, 67, 0, 14, 46, 9, 21, 54]$

 RMQ(3, 7) = 6

After preprocessing, a range minimum query is given by $RMQ(i, j)$

the output is the location of the smallest element in $A[i, j]$

<table>
<thead>
<tr>
<th>Solution 1</th>
<th>Solution 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(n)$ space</td>
<td>$O(n \log n)$ space</td>
</tr>
<tr>
<td>$O(n)$ prep time</td>
<td>$O(n \log n)$ prep time</td>
</tr>
<tr>
<td>$O(\log n)$ query time</td>
<td>$O(1)$ query time</td>
</tr>
</tbody>
</table>

**Low-resolution RMQ**

**Key Idea** replace $A$ with a smaller, 'low resolution' array $H$

and many small arrays $L_0, L_1, L_2, \ldots$ 'for the details'

$A$

$H$

Preprocess the array $H$ (which has length $\tilde{n} = \frac{n}{\log n}$) to answer RMQs...

Recall...

**Solution 2 on $A$**

$O(n \log n)$ space

$O(n \log n)$ prep time

$O(1)$ query time

**Solution 2 on $H$**

$O(\tilde{n} \log \tilde{n})$ space

$O(\tilde{n} \log \tilde{n})$ prep time

$O(1)$ query time

**Total space** $= O(n) + O(\tilde{n} \log \tilde{n} \log \log \tilde{n}) = O(n \log \log n)$

**Total prep. time** $= O(n \log \log n)$
Low-resolution RMQ

Key Idea: replace \( A \) with a smaller, ‘low resolution’ array \( H \) and many small arrays \( L_0, L_1, L_2, \ldots \) ‘for the details’

\[
\tilde{n} = \frac{n}{\log n}
\]

How do we answer a query in \( A \)?

- Do at most one query in \( H \)...
- and one query in at most two different \( L_i \) (here we query \( L_1 \) and \( L_2 \))
- then take the smallest

This takes \( O(1) \) total query time

Solution 3

\( O(n \log \log n) \) space \( \quad O(n \log \log n) \) prep time \( \quad O(1) \) query time

Range minimum query summary

Preprocess an integer array \( A \) (length \( n \)) to answer range minimum queries...

\[
\begin{array}{ccccccccccccccc}
2 & 3 & 7 & 8 & 7 & 5 & 1 & 0 & 1 & 3 & 6 & 7 & 1 & 4 & 0 & 1 & 2 & 1 & 5 & 4
\end{array}
\]

After preprocessing, a range minimum query is given by \( \text{RMQ}(i, j) \)

the output is the location of the smallest element in \( A[i, j] \)

Solution 1

\( O(n) \) space \( \quad O(n) \) prep time \( \quad O(1) \) query time

Solution 2

\( O(n \log n) \) space \( \quad O(n \log n) \) prep time \( \quad O(1) \) query time

Solution 3

\( O(n \log \log n) \) space \( \quad O(n \log \log n) \) prep time \( \quad O(1) \) query time

Can we do \( O(n) \) space and \( O(1) \) query time? Yes... but not until next lecture