List Order Maintenance

$x \rightarrow y \rightarrow z \rightarrow w$

**Query:** Is $x$ before $y$?  
Is $\text{label}(x) < \text{label}(y)$?

**Update:** Insert $x_1$ after $y$  
$\text{label}(y) < \text{label}(x_1) < \text{label}(z)$

List Order Maintenance

$x \rightarrow y \rightarrow z \rightarrow w$

Insert($y, x_1$)
Insert($x_1, x_2$)
Insert($x_2, x_3$)?? Needs relabeling…

List Order Maintenance

**Query:** $O(1)$ w.c.

**Insert:** $O(1)$ amortized


---

- Insert($y, x_1$)
- Insert($x_1, x_2$)
- Insert($x_2, x_3$) Needs relabeling…
Persistence

How to keep the history of a data structure?
How to “move” data without replication?

The DAM (Disk Access Model)

- Counting blocks between 2 levels of memory *(cost measure for complexity)*
- Models the main bottleneck
- Very Successful I (simplicity)

**Limitations**
- Parameters $B$ and $M$ are known beforehand
- It does not say anything about multilevel hierarchical memories
- Does not support dynamic $M$

Aggarwal and Vitter 1988
Focus on dynamic B-trees (although it works for any external memory structure under certain general assumptions):

1. An ephemeral B-tree maintains the most recent set of elements
2. A partially persistent B-tree maintains all history of the set of elements as they change
3. A fully persistent B-tree allows for alternate histories
## Previous Work – Our Result

<table>
<thead>
<tr>
<th></th>
<th>Update Cost</th>
<th>Range Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best Previous Result</td>
<td>$O(\log^2_B n)$</td>
<td>$O\left(\left(\log_B n + \frac{t}{B}\right)\log_B m\right)$</td>
</tr>
<tr>
<td>Our Result</td>
<td>$O(\log_B n + \log_2 B)$</td>
<td>$O\left(\log_B n + \frac{t}{B}\right)$</td>
</tr>
</tbody>
</table>

Not optimal – related to order maintenance
Our Solution

Search(version $i$, element $x$)

What is the nearest version to $i$ in $v$?
Cost: $\log_2 B$

What is the nearest version to $i$ in $u$?
Cost: $\log_2 B$

We make is $O(1)$ by using ideas from fractional cascading
Range Maxima in Secondary Memory

Ideas

1. A $B$-tree holds all points sorted by $x$-coordinate.

2. The points in each leaf are structured as a Priority Queue with attrition based on the $y$-coordinate.

3. For each internal node we persistently concatenate all the PQAs from right to left.

4. All the non-attrited points of the root are the maxima
Looking Deeper...

Replication is very expensive. Persistence comes to the rescue.

PQAs at the children are destroyed!!!
Storing the History of Graphs


Query: What is the impact of Aristotle (in terms of some measure of centrality) during the 18th century?
Previous Solutions: Snapshots and Logging

**Snapshots:** The network as it is at some particular time instance.

**Logging:** Keeping the update operations between successive snapshots.

Why not use Persistence 😊?
A GOOD STUDENT ALWAYS TRIES TO BEST HIS TEACHER. A GOOD TEACHER ASPIRES HIS STUDENTS TO DO SO. (IN OUR CASE WE HAVE A VERY LONG WAY TO GO 😊)

Loneliness, 1982