4. Dynamic Setting

Addition and Deletion

M: the set of objects in the give time after a series of addition and deletion

Remark

- $H(M)$ does no depend on the insertion order of M
- The history and the search structure depend on the insertion order

General Idea

- For each object in M, we randomly choose a priority (a real number) from the uniform distribution on the interval $(0, 1)$.
- Shuffle (or a priority order) on M is the ordering of objects of M according to the increasing priorities
- Shuffle is exactly the insertion order
- $S_k = k^{\text{th}}$ object in the shuffle, and $M^i = \{S_1, S_2, \ldots, S_i\}$
- $\widetilde{H}(\text{shuffle}(M^i))$ is the history of $H(\text{shuffle}(M^1))$, $H(\text{shuffle}(M^2))$, ..., $H(\text{shuffle}(M^i))$

4. Random Binary Tree of Quick-Sort

$M =$	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
Value	23	11	37	47	29	3	7	19
Priority	0.1	0.3	0.4	0.5	0.6	0.8	0.9	0.95

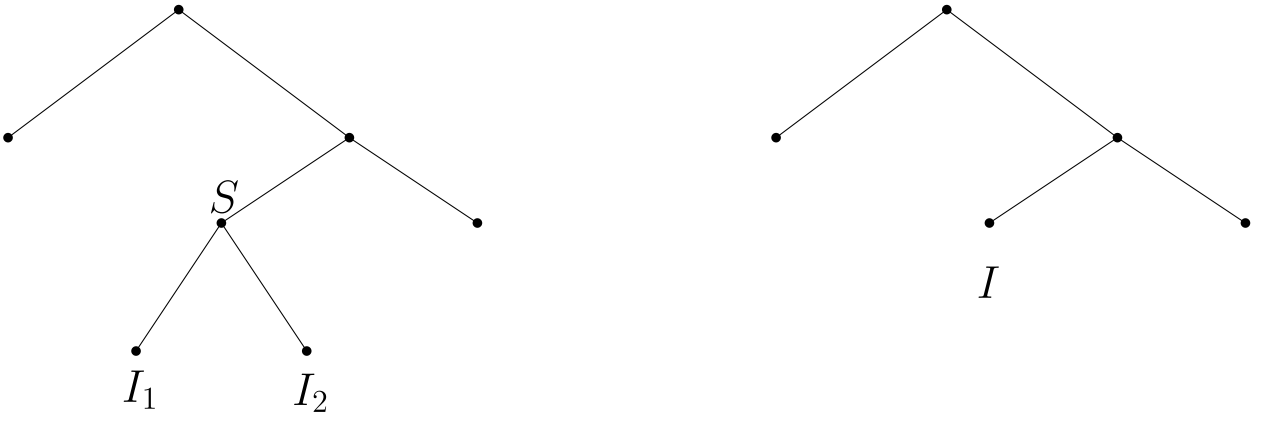
We want to delete S from M

Fact

If a point v is a descendent of a point u, u's priority is lower than v's

Case 1: S is a leaf in the sense that its two sons are both intervals

- Combine its two sons into one interval
- replace S with I

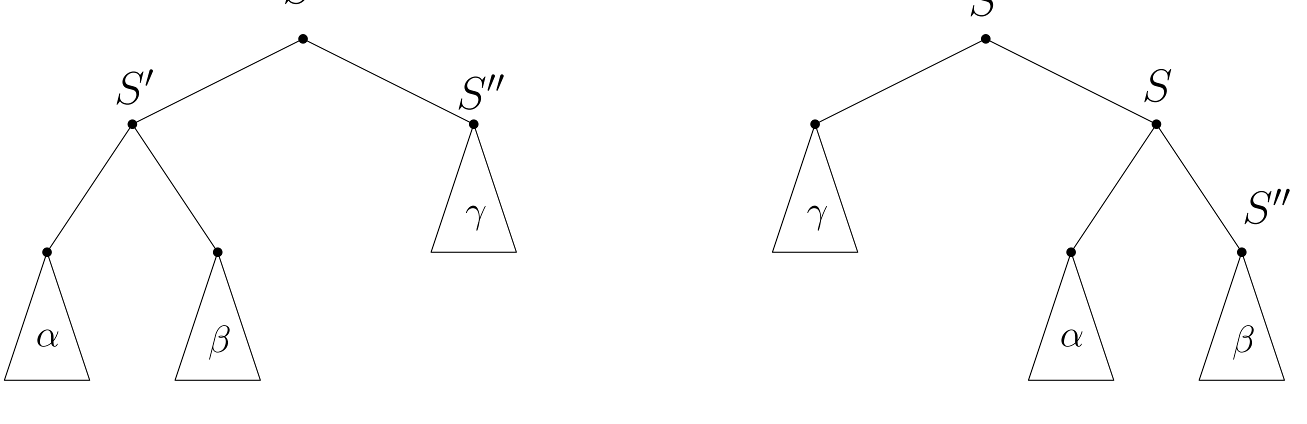


Case 2: S is an internal node in the sense that at least one of its sons is a point

- Move S to become a leaf and delete it
  - Increase S's priority step by step to prepared S is inserted later
  - When S's priority is higher than one of its sons, a rotation will happen and move S downward one level

Rotation:

Assume S has two sons S' and S'', S' has low priority than S''



Repeatedly performing rotations will bring S to the bottom such that we can delete it.

The expected number of rotations is at most the expected depths of the binary tree, which is $O(\log n)$

Adding S

- use $\widetilde{H}(M)$ to locate S
- Split the interval contains S
- Assume S' to be the original parent of the interval. Let S' be the parent of S and S be the parent of the two new intervals
- Assign S a priority higher than S'

Another viewpoint of rotation:
Deleting S from shuffle(M) can be carried out by moving S higher one by one.

l is the number of points with priorities higher than S

$M(i)$ is the priority-ordered set obtained from M by moving S higher in the order by i places.

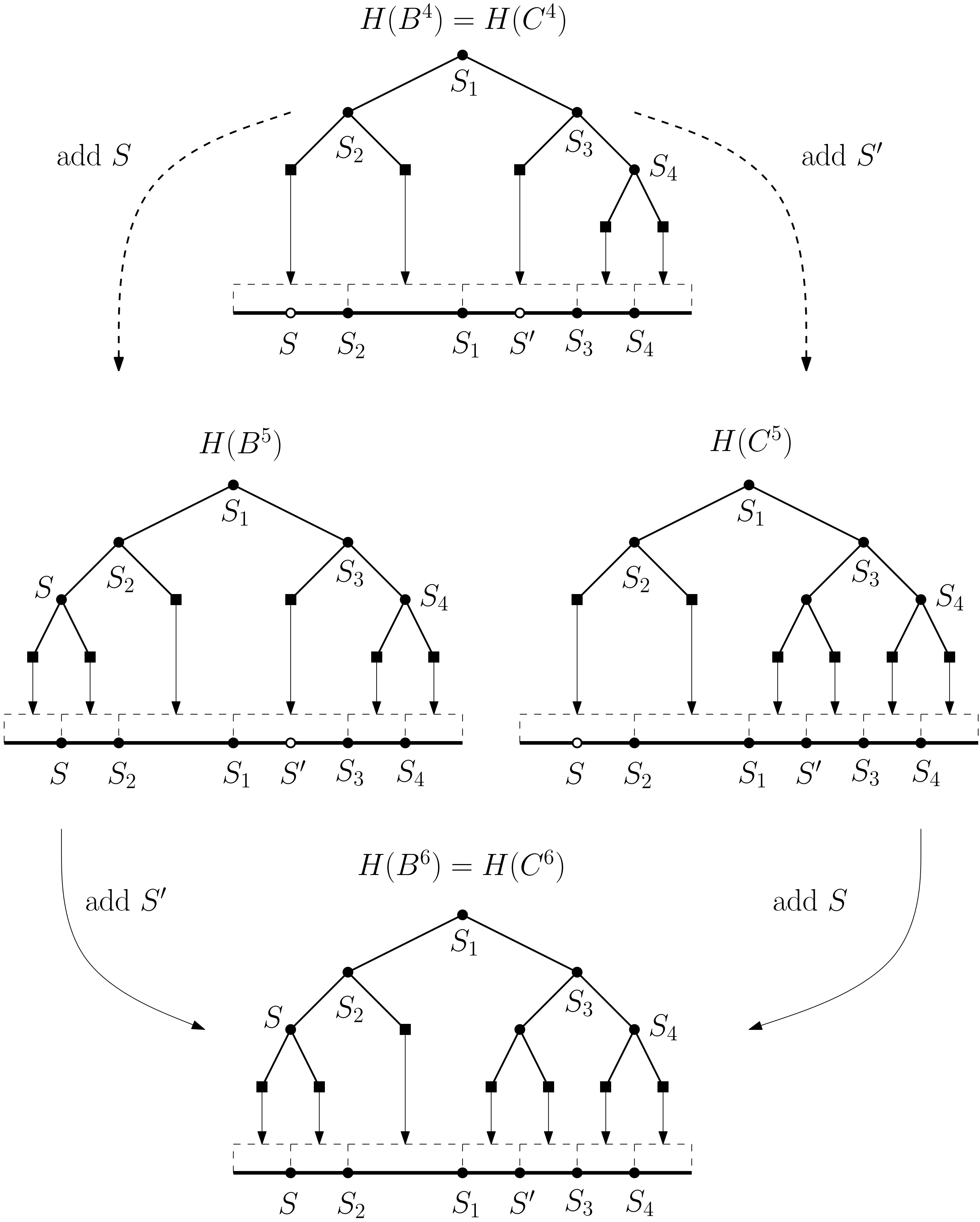
shuffle($M(0)$), shuffle($M(1)$), . . ., shuffle($M(l)$)

$B = M(i-1)$ and $C = M(i)$
Let S' be the point just after S in the priority order B
$j = m - l + i - 1$, i.e., $H(C) = H(C^j)$ and $H(B) = H(B^j)$

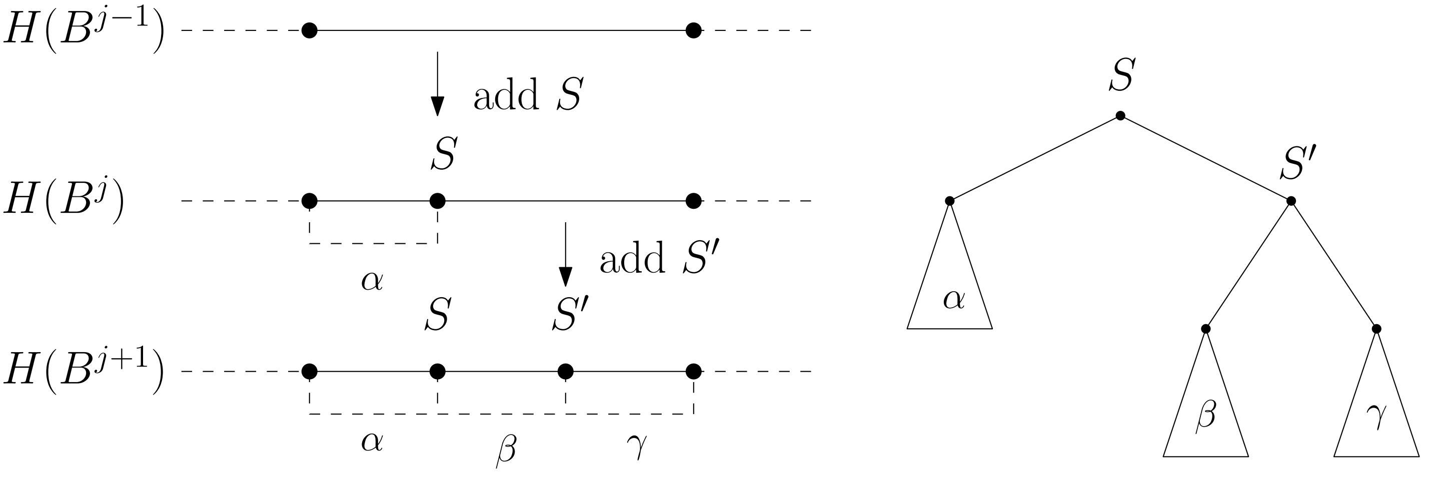
$$
\begin{array}{ccccccccc}
\longrightarrow & H(B^{j-1}) & \xrightarrow{S} & H(B^j) & \xrightarrow{S'} & H(B^{j+1}) & \longrightarrow & H(B^{j+2}) & \longrightarrow \\
 & = & & & & = & & = & \\
\longrightarrow & H(C^{j-1}) & \xrightarrow{S'} & H(C^j) & \xrightarrow{S} & H(C^{j+1}) & \longrightarrow & H(C^{j+2}) & \longrightarrow
\end{array}
$$

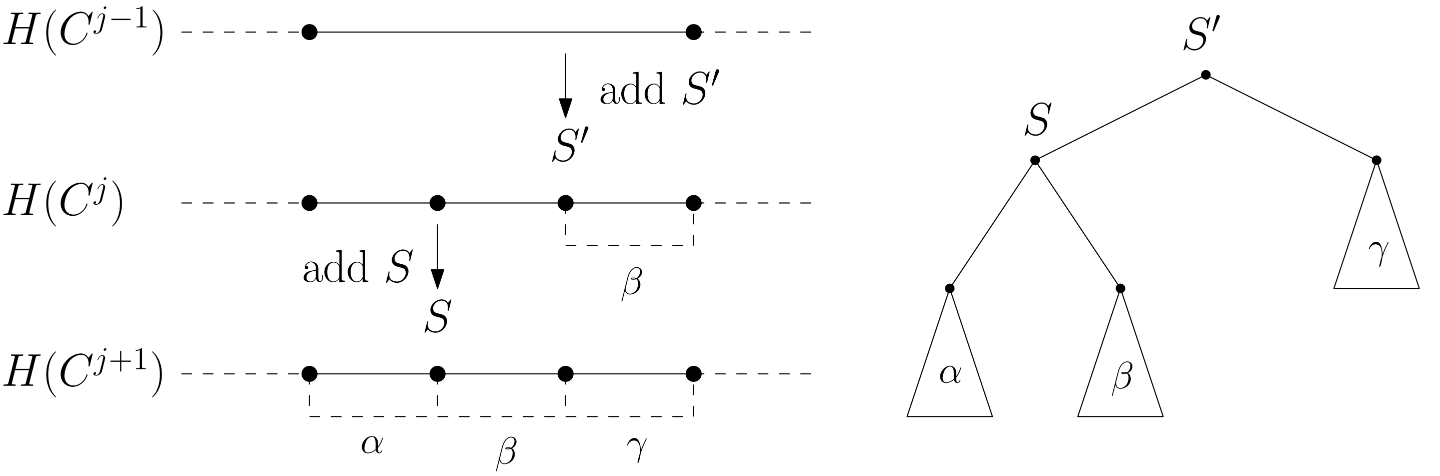
If $H(B) = H(C)$, this operation is free.

- S and S' is contained in different intervals of $H(B^{j-1})$



If we give S a priority above that of S', a right rotation can reflect the new shuffle on M





4.2 Trapezoidal Decomposition