Soft-heaps according to Kaplan and Zwick

A node v stores:

- $\ell(v)$, $r(v)$ – left and right child
- $\mathit{rank}(v)$ – rank; never changes; ranks of children are smaller by 1
- $\mathit{list}(v)$ – a list of items stored in this node
- $\mathit{ckey}(v)$ – a key common to all keys in $\mathit{list}(v)$
- $\mathit{size}(v)$ – planned size of $\mathit{list}(v)$

We build trees out of the nodes:

- ckeys of nodes are heap-ordered
- $\mathit{rank}(T)$ and $\mathit{ckey}(T)$ inherited from the root node

A heap $\mathcal{H}$ contains a list of trees in order of increasing rank. A rank of the heap is a maximum of tree ranks. For each tree T, we store:

- $\mathit{sufmin}(T)$ – pointer to the tree with minimum ckey following T

Setup of parameters:

- $r = \lceil \log_2(1/\varepsilon) \rceil + 5$
- $s_k = 1$ for $k \le r$, else $s_k = \left\lceil \frac{3s_{k-1}}{2} \right\rceil$
- $\mathit{size}(v) = s_k$, where $k = \mathit{rank}(v)$

Observation: $(3/2)^{k-r} \le s_k \le 2 \cdot (3/2)^{k-r} - 1$ for $k \ge r$.

Sift(v):

1. While $|\mathit{list}(v)| < \mathit{size}(v)$ and v is not a leaf:
2. If $\ell(v) = \emptyset$ or $\mathit{ckey}(\ell(v)) > \mathit{ckey}(r(v))$: $\ell(v) \leftrightarrow r(v)$.
3. Move all items from $\mathit{list}(\ell(v))$ to $\mathit{list}(v)$.
4. $\mathit{ckey}(v) \leftarrow \mathit{ckey}(\ell(v))$.
5. If $\ell(v)$ is a leaf, remove it; else $\mathit{Sift}(\ell(v))$.

Invariant L: $\mathit{size}(v)/2 \le |\mathit{list}(v)| \le 3 \cdot \mathit{size}(v)$ for nodes of rank at least r; otherwise $|\mathit{list}(v)| \ge 1$.

Invariant R: #nodes of rank $k \le n/2^k$.

Invariant C: #corrupted items $\le \varepsilon n$.

Potential:

- Heap of rank k contributes $k + 1$.
- A tree with root x contributes $(r + 2) \cdot \mathit{del}(x)$, where $\mathit{del}(x)$ is the number of items deleted from $\mathit{list}(x)$ since the previous call to Sift or creation of the root.
- A root of rank k contributes $k + 7$.
- Every other node contributes 1.