Vladislav Martyniuk
martyvla@fit.cvut.cz
presented:

QuickHeapsort: Modifications and Improved Analysis

Volker Diekert, ArminWeiß
https://www.researchgate.net/publication/230877469_QuickHeapsort_Modifications_and_Improved_Analysis

---

Definitions

QuickHeapsort

- **Abstract** QuickHeapsort is a combination of Quicksort and Heapsort. It is based on Katajainen's idea for Ultimate Heapsort. Let the array is partitioned into two parts by some pivot element. QuickHeapsort sorts the smaller part like HeapSort and calls itself recursively for the larger part, only.
- Simple **in-place modification** of QuickHeapsort saves $0.75n$ comparisons. Using **n extra bits** only we can bound the expected number of comparisons to $nlog_2 n - 0.997n + o(n)$.
- QuickHeapsort variants: Basic QuickHeapsort, Improved QuickHeapsort, QuickHeapsort with bit arrays. With median of 3 and $\sqrt{n}$.
- Others algorithms in competition with QuickHeapsort: Quicksort, Ultimate Heapsort, Bottom-Up-Heapsort, MDR-Heapsort.

Two-layer-heap

- A **two-layer-min-heap** is an array $A[1..n]$ of n elements together with a partition (G, R) of $1, ..., n$ into *green* and *red* elements such that for all $g \in G$, $r \in R$ we have $A[g] \leq A[r]$.

⇒ the green elements g satisfy the heap condition $A[g] \leq minA[2g], A[2g+1]$.
⇒ if r is red, then $2r$ and $2r+1$ are red, too.

- **Two-layer-maxheaps** are defined analogously.

Algorithm

- ChoosePivot: It returns an element p of the array.
- PartitionReverse: It returns an index k and rearranges the array A so that $p = A[k]$, $A[i] \geq A[k]$ for $i < k$ and $A[i] \leq A[k]$ for $i > k$ using $n-1$ comparisons.
- ConstructMaxHeap: Constructs a max-heap on the input array.

Theorems

Theorem 1. The expected number $\mathbb{E}[T(n)]$ of comparisons by basic (resp. improved) QuickHeapsort with pivot as median of k randomly selected elements on an input array of size n satisfies $\mathbb{E}[T(n)] \leq n \lg n + c_k n + o(n)$ with c_k as follows:

k	c_k basic QHS variant	c_k improved QHS variant
1	+2.72	+1.97
3	+1.92	+1.17
$f(n)$	+0.72	-0.03

Proposition 1. $T_{cstr}(n) \leq 1.625n + o(n)$.

Theorem 2. $T_{ext}(n) \leq n \cdot (\lfloor \lg n \rfloor - 3) + 2\{n\} + O(\lg^2 n)$.

Lemma 1. Let $x \geq y > \delta \geq 0$. Then we have the inequalities: $F(x) + F(y) \leq F(x+\delta) + F(y-\delta)$ and $F(x) + F(y) \leq F(x+y)$.

Lemma 2. Let $1 \leq v \in \mathbb{R}$. For all sequences $x_1, x_2, ..., x_t$ with $x_i \in R^{>0}$, which are valid w.r.t. v, we have:

$$\sum_{i=1}^{t} F(x_i) \leq \sum_{i=1}^{\lfloor \lg v \rfloor} F(\frac{v}{2^i})$$

.

Lemma 3.

$$\sum_{i=1}^{\lfloor \lg n \rfloor} F(\frac{n}{2^i}) \leq F(n) - 2n + O(\lg n)$$

.

Corollary 1. We have $T_{ext}(n) \leq n \lg n - 2.9139n + O(\lg^2 n)$.

Lemma 4. Let $0 < \delta < \frac{1}{2}$ and $\alpha = 4(\frac{1}{4} - \delta^2) < 1$. If we choose the pivot as median of $2c+1$ elements such that $2c+1 \leq \frac{n}{2}$ then we have

$$Pr[pivot \leq \frac{n}{2} - \delta n] < (2c+1)\alpha^c$$

.

Theorem 3. Let $f \in \omega(1) \cap o(n)$ with $1 \leq f(n) \leq n$ and let the pivot be chosen as median of $f(n)$ randomly selected elements. Then the expected number of comparisons used in all recursive calls of partitioning satisfies

$$\mathbb{E}[T_{part}(n)] \leq 2n + o(n)$$

.

Corollary 2. Let $f \in \omega(1) \cap o(n)$ with $1 \leq f(n) \leq n$. When implementing Quickselect with the median of $f(n)$ randomly selected elements as pivot, the expected number of comparisons is $2n + o(n)$.

Theorem 4. Let $f \in \omega(1) \cap o(n)$ with $1 \leq f(n) \leq n$ e.g., $f(n) = \sqrt{n}$, and let $\mathbb{E}[T(n)]$ be expected number of comparisions by QuickHeapsort using the CompareArray with the improvement and the RedGreenArray on a fixed input array of size n. Choosing the pivot as median of $f(n)$ randomly selected elements in time $O(f(n))$, we have

$$\mathbb{E}[T(n)] \leq n \lg n - 0.997n + o(n)$$

.