Median of k-medians

k odd

$$T_A(n) = O(n) + T_O\left(\frac{n}{k}\right), \qquad O(n) = \phi_k \cdot \frac{n}{k} \tag{1}$$

$$T_O(n) = T_A(n) + O(n) + T_O\left(\left(1 - \frac{1}{2}\cdot\frac{k+1}{2k}\right)\cdot n\right), \qquad O(n) \text{ independent of } k \tag{2}$$

ϕ_k in (1) depends on what algorithm you use to determine the (exact!) median of the groups of k.
AppMedian() calls OrderStat() (with $K = {}^{n}/_{2k}$)
$O(n)$ independent of k in second equation (2) comes from calling the Partition() algorithm.
k even

$$T_A(n) = O(n) + T_O\left(\frac{n}{k}\right), \qquad O(n) = \phi_k \cdot \frac{n}{k}$$

$$T_O(n) = T_A(n) + O(n) + T_O\left(\left(1 - \frac{1}{4}\right)\cdot n\right), \qquad O(n) \text{ independent of } k$$

For simplicity ignore $O(n)$ independent of k below in (3) and (4).
k odd

$$T_O(n) = O(n) + T_O\left(\frac{n}{k}\right) + T_O\left(\left(1 - \frac{k+1}{4k}\right)\cdot n\right), \qquad O(n) = \phi_k \cdot \frac{n}{k} \tag{3}$$

$$T_O(n) \sim n \cdot \frac{\phi_k}{k\left(1 - \frac{1}{k} - \left(1 - \frac{k+1}{4k}\right)\right)} = n \cdot \frac{\phi_k}{k\left(1 - \left(\frac{3}{4} + \frac{3}{4k}\right)\right)}$$

k even

$$T_O(n) = O(n) + T_O\left(\frac{n}{k}\right) + T_O\left(\left(1 - \frac{1}{4}\right)\cdot n\right), \qquad O(n) = \phi_k \cdot \frac{n}{k} \tag{4}$$

$$T_O(n) \sim n \cdot \frac{\phi_k}{k\left(1 - \frac{1}{k} - \left(1 - \frac{1}{4}\right)\right)} = n \cdot \frac{\phi_k}{k\left(1 - \left(\frac{3}{4} + \frac{1}{k}\right)\right)}$$

So

$$T_A(n) \sim \phi_k \cdot \frac{n}{k} + n \cdot \frac{\phi_k}{k^2\left(1 - \left\{ \begin{array}{ll} \frac{3}{4} + \frac{3}{4k} & k \text{ odd} \\ \frac{3}{4} + \frac{1}{k} & k \text{ even} \end{array} \right\}\right)} = \lambda_k \cdot n$$

with

$$\lambda_k = \frac{\phi_k}{k} + \frac{\phi_k}{k^2\left(1 - \left\{ \begin{array}{ll} \frac{3}{4} + \frac{3}{4k} & k \text{ odd} \\ \frac{3}{4} + \frac{1}{k} & k \text{ even} \end{array} \right\}\right)}.$$

For $T_A(n) \sim O(n)$, we need $\{\ldots\} < 1$ in the denominator, so $k \geq 5$! But why $k = 5$?

How about constants? Depends on what algorithm you use to determine the k-group median.

Example 1. Use bubble sort $\phi_k = k^2$. Then λ_k becomes

$$\begin{array}{l|lcl} k=5 & 5 + \frac{25}{25\left(1-\frac{9}{10}\right)} & = & 15 \\ k=6 & 6 + \frac{36}{36\left(1-\frac{11}{12}\right)} & = & 18 \\ k=7 & 7 + \frac{49}{49\left(1-\frac{6}{7}\right)} & = & 14 \\ k=8 & 8 + \frac{64}{64\left(1-\frac{7}{8}\right)} & = & 16 \\ k=9 & 9 + \frac{81}{81\left(1-\frac{5}{6}\right)} & = & 15 \end{array}$$

Example 2. Use merge/quicksort (w/ linear time median) $\phi_k = k \cdot \log k$. (I use below log = ln.)

$$\begin{array}{l|lcl} k=5 & 3\log 5 & \approx & 4.82 \\ k=6 & 3\log 6 & \approx & 5.36 \\ k=7 & 2\log 7 & \approx & 3.89 \\ k=8 & 2\log 8 & \approx & 4.16 \\ k=9 & \frac{5}{3}\log 9 & \approx & 3.66 \end{array}$$

This suggests that larger k could give better performance (test it!) But this will not decrease continuously (why?)

Example 3. "Use" linear time (5-group) median $\phi_k = k$. Then

$$\begin{array}{l|l} k=5 & 3 \\ k=6 & 3 \\ k=7 & 2 \\ k=8 & 2 \\ k=9 & \frac{5}{3} \end{array}$$

Then constants will decrease continuously (why?) BUT our linear time median does not give an exact median, only an approximate (30-70) one. Thus we cannot use it in this way.

Note that the quality of median approximation output is different when k is different. The median of k-medians will give a

$$(\varepsilon_k, 1-\varepsilon_k)$$

approximation for

$$\varepsilon_k = \left\{ \begin{array}{ll} \frac{k+1}{4k} & k \text{ odd} \\ \frac{1}{4} & k \text{ even} \end{array} \right\}$$

So the median of 6-medians is not only slower than the median of 5-medians, but it also gives a worse (25-75) output, which will slow down quicksort as well.

To see the complexity $\mu_k \cdot n \log n$ of quicksort, we calculate

$$\mu_k = \frac{-\lambda_k}{\varepsilon_k \log(\varepsilon_k) + (1-\varepsilon_k)\log(1-\varepsilon_k)}$$

Example 4. Best ϕ_k we can use is w/ merge/quicksort $\phi_k = k \cdot \log k$.

$$\begin{array}{r|ll} k=5 & \frac{-3\log 5}{0.3\log 0.3+0.7\log 0.7} & \approx 7.90 \\ k=6 & \frac{-3\log 6}{0.25\log 0.25+0.75\log 0.75} & \approx 9.13 \\ k=7 & \frac{-2\log 7}{2/7\log 2/7+5/7\log 5/7} & \approx 6.51 \\ k=8 & \frac{-2\log 8}{0.25\log 0.25+0.75\log 0.75} & \approx 7.06 \\ k=9 & \frac{-5/3\log 9}{5/18\log 5/18+13/18\log 13/18} & \approx 6.20 \end{array}$$

Again, μ_k will not decrease forever, but it provides a lot of evidence that $k=5$ is far from the best.

Problem 5. Implement and test for which k is qsort w/ median of k-medians (via qsort in the groups of k) the fastest?

IMPROVEMENT FOR k EVEN: *Median of alternate k-medians*
Use the $k/2$-th median element in the odd groups of k and $k/2+1$-st element in the even groups (or vice versa). Then formulas for odd k will be approximated for even k as well. But worst case remains the same, since we cannot know in advance if the k-groups whose median is larger than the median of k-medians are almost all even or odd.

Problem 6. Can you make the so improved median of 6-medians faster than the median of 5-medians? How about $k=4$ then?