A proof that Accelerated FixHeap run on an almost – heap of n nodes will perform no more than $\lfloor \lg n \rfloor + \lfloor \lg \lg n \rfloor + 1$ comparisons.

Let $C_{AccFixHeap}(n)$ be the number of comparison that the Accelerated FixHeap (call it AFH) performs in the worst case while fixing an almost– heap H on n nodes.

AFH will demote the root R of H, if necessary, down along the path P of the larger child. The length of P is no more than $\lfloor \lg n \rfloor$, so AFH will perform no more than
$\lfloor \lg n \rfloor$
comparisons (one comp per level) just to find P while demoting R.

This upper bound remains an upper bound even if R has been accidentally demoted too far (say, to the level L, with L comparisons since one comparison is performed for each level) and must be promoted up one or more levels. In such a case, R will be promoted no further then one level below the level at which R was compared previously to a node in P. The distance (up) of such promotion, measured in the number of levels passed, is not larger than the distance (down) $\lfloor \lg n \rfloor - L$ from L to the last level of H. (Excercise: Prove it!) Since only one comparison per level is performed while promoting R, the total number of comparisons for all demotions and subsequent promotions must be not larger than $L + \lfloor \lg n \rfloor - L = \lfloor \lg n \rfloor$.

Now comes the Binary Search part.

There are $\lfloor \lg n \rfloor + 1$ nodes along P, one of wich is the root R of H. So, there are $\lfloor \lg n \rfloor$ remaining nodes along P. All these remaining nodes are ordered decreasingly AFH needs to find the level of H at which to insert R into P using the Binary Search.

The worst–case number of comparisons performed by Binary Search on m ordered elements is $\lfloor \lg m \rfloor + 1$.

Since there are $m = \lfloor \lg n \rfloor$ nodes (all nodes of P except for the root R), the worst– case number of comparisons that Binary Search will perform on those nodes is $\lfloor \lg m \rfloor + 1 =$
$\lfloor \lg \lfloor \lg n \rfloor \rfloor + 1$.

Therefore, the number of comparisons $C_{AccFixHeap}(n)$ performed by AFH will be not greater than $\lfloor \lg n \rfloor + \lfloor \lg \lfloor \lg n \rfloor \rfloor + 1$, that is,

$$C_{AccFixHeap}(n) \le \lfloor \lg n \rfloor + \lfloor \lg \lfloor \lg n \rfloor \rfloor + 1.$$

Let's simplify $\lfloor \lg \lfloor \lg n \rfloor \rfloor$.

$$\lfloor \lg \lfloor \lg n \rfloor \rfloor = \max\{i : i \le \lg \lfloor \lg n \rfloor\} = \max\left\{i : 2^i \le 2^{\lg \lfloor \lg n \rfloor}\right\} = \max\left\{i : 2^i \le \lfloor \lg n \rfloor\right\}.$$

Now, let's show that for any integer i,
$2^i \le \lfloor \lg n \rfloor \Leftrightarrow 2^i \le \lg n$.

Of course, if $2^i \le \lfloor \lg n \rfloor$ then $2^i \le \lg n$, simply because $\lfloor \lg n \rfloor \le \lg n$.

Let's assume $2^i \le \lg n$.

Since i is an integer, 2^i is an integer, too. Because $\lfloor \lg n \rfloor$ is the largest integer m that satisfies $m \le \lg n$, we must have $2^i \le m$. In other words, $2^i \le m = \lfloor \lg n \rfloor$. Hence, $2^i \le \lfloor \lg n \rfloor$.

This way we proved $2^i \le \lfloor \lg n \rfloor \Leftrightarrow 2^i \le \lg n$.

Therefore

$$\max\left\{i : 2^i \le \lfloor \lg n \rfloor\right\} = \max\left\{i : 2^i \le \lg n\right\} = \max\left\{i : \lg 2^i \le \lg \lg n\right\} =$$
$$= \max\{i : i \le \lg \lg n\} = \lfloor \lg \lfloor \lg n \rfloor \rfloor.$$

Thus

$$\lfloor \lg \lfloor \lg n \rfloor \rfloor = \lfloor \lg \lg n \rfloor.$$

Hence,

$$C_{AccFixHeap}(n) \le \lfloor \lg n \rfloor + \lfloor \lg \lg n \rfloor + 1.$$

This completesthe proof.