EXERCISE FOR CSE202 – WEEK 9

1. Union-Find

The amortized complexity estimate for a sequence of m union or find operations with rank and path compression can be further improved. Recall that the simple bound $T(m,n,r) \leq nr$ was used in an intermediate step to compute a better bound $T(m,n,r) \leq m + 2n\log^* r$. The basic observation is that we can improve the estimate by using this better bound instead of the simple one.

Question 1. *Using this idea show that the amortized complexity of that algorithm is actually $O(m\log^*\log^* n)$ array accesses ($m \geq n$). Indicate for which value of n this function $\log^*\log^* n$ becomes larger than 3.*

Solution. The bound we have on $T(F,C)$ is $m+2n\log^* r$. Using this bound for the high forest gives

$$T(F_+,C_+) \leq m_+ + 2\frac{n}{2^s}\log^* r.$$

Thus, from the inequality at the bottom of slide 47,

$$T(F,C) \leq T(F_-,C_-) + 2m_+ + n + 2\frac{n}{2^s}\log^* r$$

and since $m_+ = m - m_-$,

$$T(F,C) - 2m \leq T(F_-,C_-) - 2m_- + n + 2\frac{n}{2^s}\log^* r.$$

Choosing $s = \lceil \log_2 \log^* r \rceil$, the last summand becomes smaller than $2n$, whence

$$T(F,C) - 2m \leq T(F_-,C_-) - 2m_- + 3n.$$

where now F_- is a forest all whose nodes have rank at most $\log_2\log^* r$. Iterating this construction on this forest and so on $\log^*\log^* r$ times gives

$$T(F,C) \leq 2m + 3n\log^*\log^* r = O(m\log^*\log^* n).$$

The largest value of k such that $\log^*\log^* k = 3$ satisfies $\log^* k = 16$, which means that k is obtained by iterating 16 times the map $x \mapsto 2^x$ starting from $x = 1$. This is a number that is unimaginably large (and so is its number of digits). This, plus 1, is the value where this function becomes larger than 3. □