Parallel comparison

- sketch(node) = 1 sketch(x_0) 1 sketch(x_1) ··· 1 sketch(x_{k-1})
- sketch(q)^k = 0 sketch(q) 0 sketch(q) ··· 0 sketch(q)
  = 0000001 000001 ··· 000001 × sketch(q)
- sketch(node) − sketch(q)^k
  = $\binom{1}{0}$***** $\binom{1}{0}$***** ··· $\binom{1}{0}$*****
- AND with 1000001 00000 ··· 100000
  → $\binom{1}{0}$00000 $\binom{1}{0}$00000 ··· $\binom{1}{0}$00000
- extra 1's in sketch(node) protect from underflow
  - remain 1 if sketch(q) ≤ sketch(x_i)
  - become 0 if sketch(q) > sketch(x_i)

⇒ $\binom{1}{0}$ bits are 0000 1111

where sketch(q) fits ↑ ↑ most significant 1 bit

(index of) Most significant 1 bit: 00010110 ↦ 4

7 6 5 4 3 2 1 0

- AC^0 operation [Andersson, Miltersen, Thorup – TCS 1999]
- instruction on many real CPUs (see Linux kernel: include/asm-*/bitops.h)
  e.g. Cray, Pentium
- possible with O(1) arithmetic & Boolean ops. [Brodnik 1993]
- easy solution with extra space:
  - lookup table on all strings of $\varepsilon \cdot w$ bits
  ⇒ $O(2^{\varepsilon w} \cdot \lg w)$ bits = $O(u^{\varepsilon} \lg\lg u)$ bits of space
  - query: $O(1/\varepsilon)$ [or $O(\lg 1/\varepsilon)$] probes to table