Longest Increasing Subsequence

Longest increasing sequence. Given a sequence of elements $c_1, c_2, \ldots, c_n$ from a totally ordered set, find the longest increasing subsequence.

Ex: 7 2 8 1 3 4 10 6 9 5

Application. Part of MUMmer system for aligning entire genomes.

Dynamic programming solution. $O(n^2)$.
LIS is a special case of edit-distance.
- $X = c_1 c_2 \ldots c_n$
- $y = \text{sorted sequence of } c_i, \text{removing any duplicates}$
- mismatch penalty = $\infty$

Patience

Patience. Deal cards $c_1, c_2, \ldots, c_n$ into piles according to two rules:
- Can't place a higher valued card onto a lowered valued card.
- Can form a new pile and put a card onto it.

Goal. Form as few piles as possible.

Patience: Greedy Algorithm

Greedy algorithm. Place each card on leftmost pile that fits.
**Patience: Greedy Algorithm**

**Greedy algorithm.** Place each card on leftmost pile that fits.

**Observation.** At any stage during greedy algorithm, top cards of piles increase from left to right.

---

**Patience-LIS: Weak Duality**

**Weak duality.** In any legal game of patience, the number of piles $\geq$ length of any IS.

**Pf.** Cards within a pile form a decreasing subsequence. An IS can use at most one card from each pile.

---

**Patience-LIS: Strong Duality**

**Strong duality.** Min number of piles = max length of an IS.

**Moreover.** Greedy algorithm finds both.

**Pf.** Each card maintains a pointer to top card in previous pile. Follow pointers to obtain an IS whose length equals the number of piles. By weak duality, both are optimal.

---

**Greedy Algorithm: Implementation**

**Efficient implementation.** $O(n \log n)$

- Use $n$ stacks to represent $n$ piles.
- Use binary search to find leftmost legal pile.

```
Create n empty stacks $S_1, \ldots, S_n$

foreach (card c in deck order) {
    $S_i \leftarrow$ binary search to find leftmost stack that fits c
    push card c on top of stack $S_i$
    make card c point to top card of stack $S_{i-1}$
}

Form LIS by following back-pointers from top card of rightmost nonempty stack
```
**Patience Sorting**

**Patience sorting.** Deal all cards; repeatedly remove smallest card.

**Theorem.** Expected number of piles is approximately $2^{n/2}$ with standard deviation around $n^{1/6}$ if deck is uniformly random.

**Remark.** An almost-trivial $O(n^{3/2})$ sorting algorithm.

**Speculation.** [Persi Diaconis] Patience sorting is the fastest way to sort a pile of cards by hand.

**Bonus Theorem**

**Theorem.** [Erdős-Szekeres, 1935] A sequence of $n^2 + 1$ distinct real numbers either has an increasing or decreasing subsequence of size $n + 1$.

**Pf.** (by pigeonhole principle)