Lecture 8: Strong Induction & Sorting

Fall 2016

Kim Bruce & Peter Mawhorter

Quiz #2

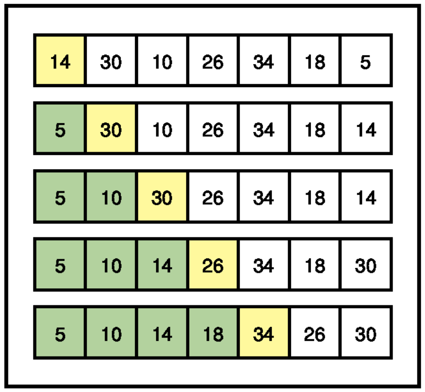
You do not have to do a proof.

Assignment 1

- Grading is not quite done.
- We'll try to provide feedback before Sunday.
- Assignment 2 is due on Sunday.

For Today

- Selection sort proof
- Strong induction
- Merge sort



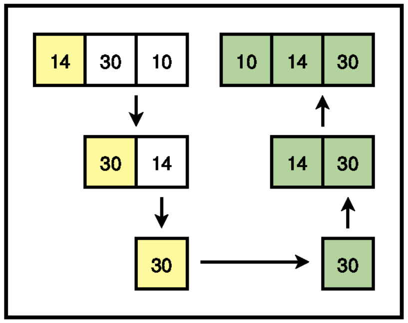
Selection sort progress.

Correctness

Can we prove that our algorithm works?

(use induction)

What must be true after each step?



Selection sort recursion.

Complexity

Can we prove that our algorithm works *quickly*?

How many operations does each `indexOfSmallest` take?

$$\sum_{i=1}^{n} i \rightarrow \frac{n(n+1)}{2}$$

Strong Induction

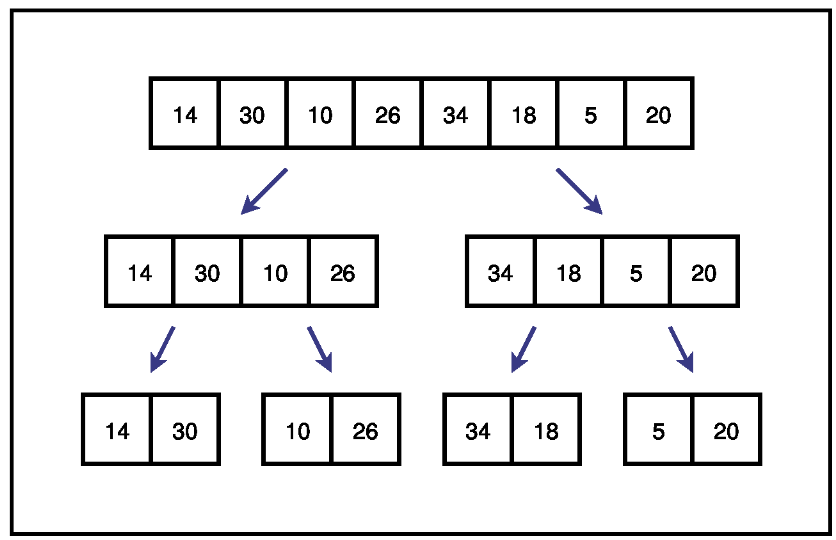
- Instead of just assuming P(k) and proving P($k+1$)…
- Assume P(k) *for all* $0 \leq k < n$ to prove P(n)
  - Use when just the previous case is not enough.

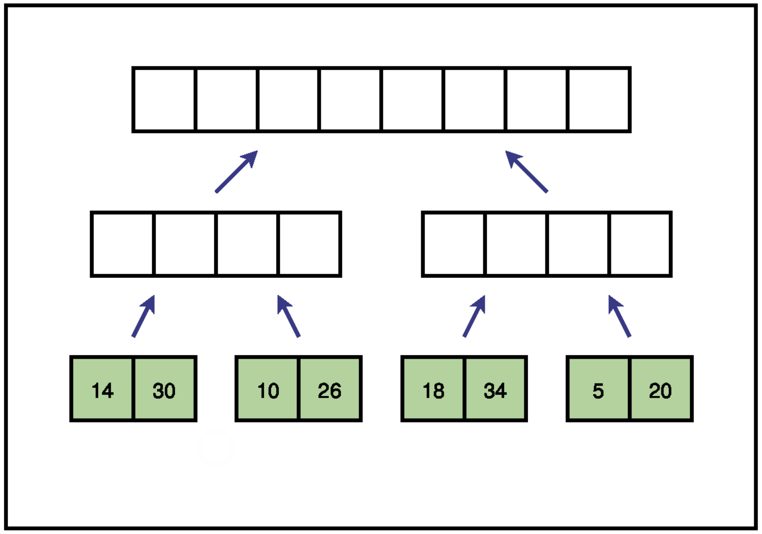
Fast Exponentiation

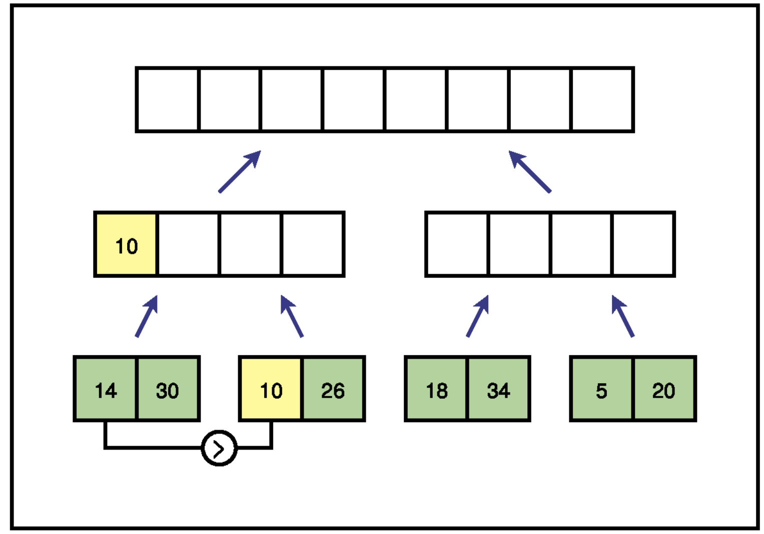
- `fastPower`(x, n) calculates x^n:
  - if $n == 0$, return 1
  - if n is even, return `fastPower`$(x * x, n/2)$
  - if n is odd, return $x \times$`fastPower`$(x, n-1)$
- Proof by induction on n:
  - Base case: $n == 0$
  - Assumption: assume `fastPower`(x, k) is x^k for all $0 \leq k < n$.
  - Inductive case: show `fastPower`(x, n) is x^n

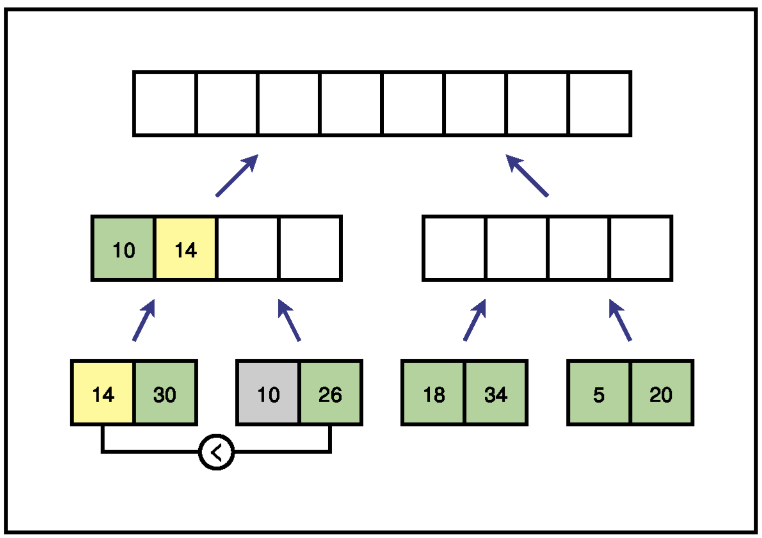
Divide-and-Conquer

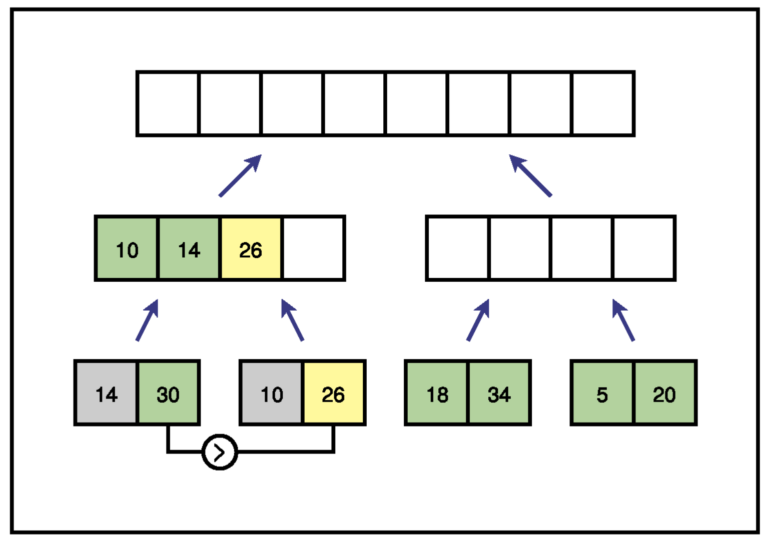
- Some problems are tough.
  - But maybe we can divide them into simpler problems.
  - …and keep going until all we have are trivial problems?
  - Now we just need to combine the solutions.
- Sounds like a job for recursion!

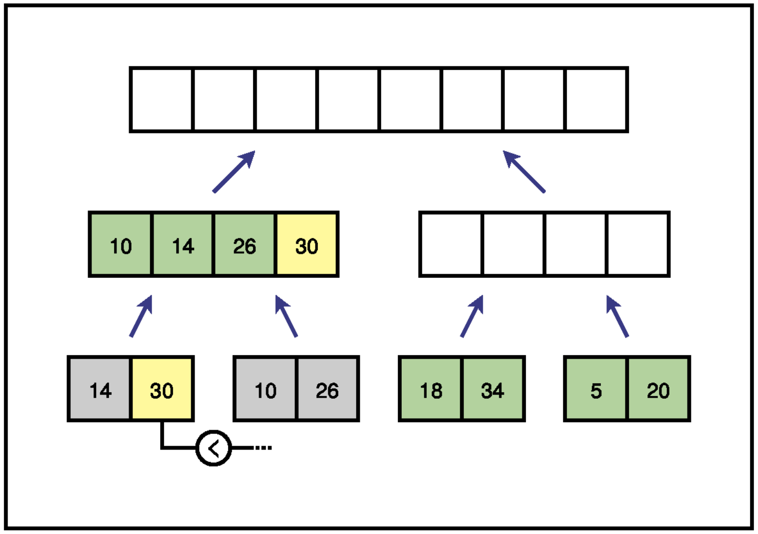


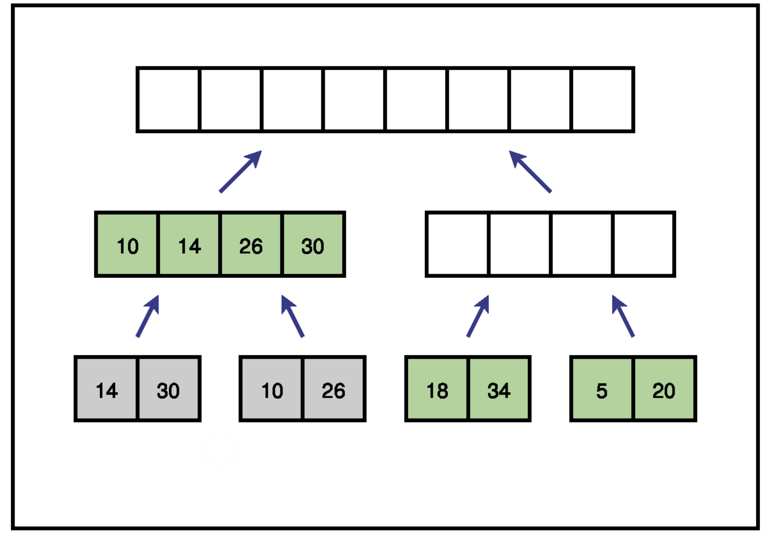

14 30 10 26 18 34 5 20

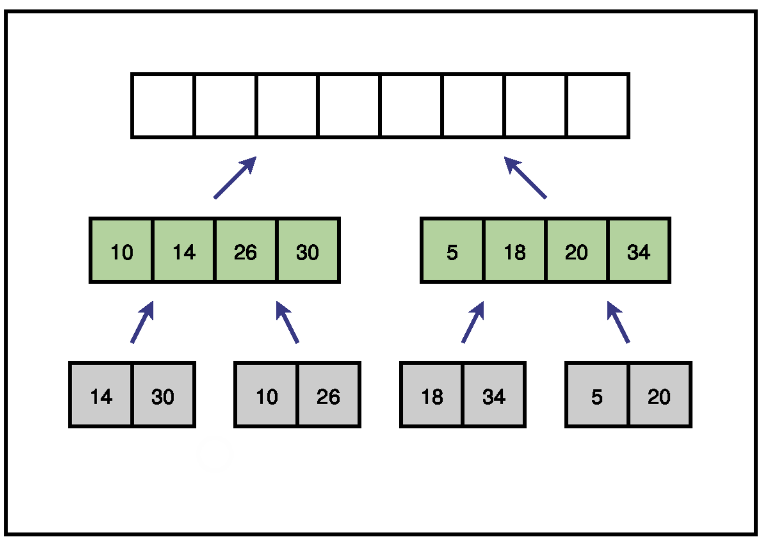

10
14 30 10 26 18 34 5 20

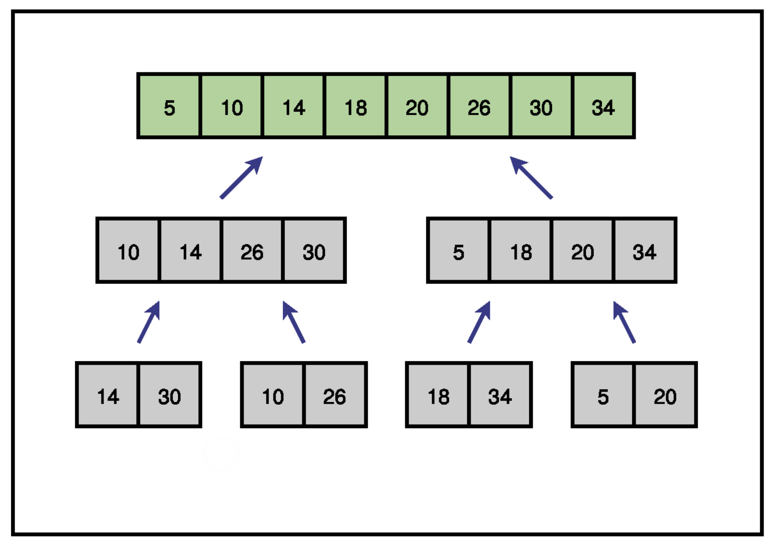

10 14
14 30 10 26 18 34 5 20


10 14 26
14 30 10 26 18 34 5 20


10 14 26 30
14 30
10 26
18 34
5 20


10 14 26 30
14 30
10 26
18 34
5 20


10 14 26 30
5 18 20 34
14 30
10 26
18 34
5 20


5 10 14 18 20 26 30 34
10 14 26 30
5 18 20 34
14 30
10 26
18 34
5 20

Mergesort Steps

1. Divide into two halves.
2. Sort each half.
3. Merge the results and return.

Correctness

Use strong induction.

Time Complexity

Use strong induction.

Summary

- MergeSort makes $f(n) = 2f(n/2) + n$ comparisons on an array of size n
- Strong induction $\rightarrow$ $f(n)$ is in $O(n \log_2 n)$