S.KIMMEL

What's the big Idea?

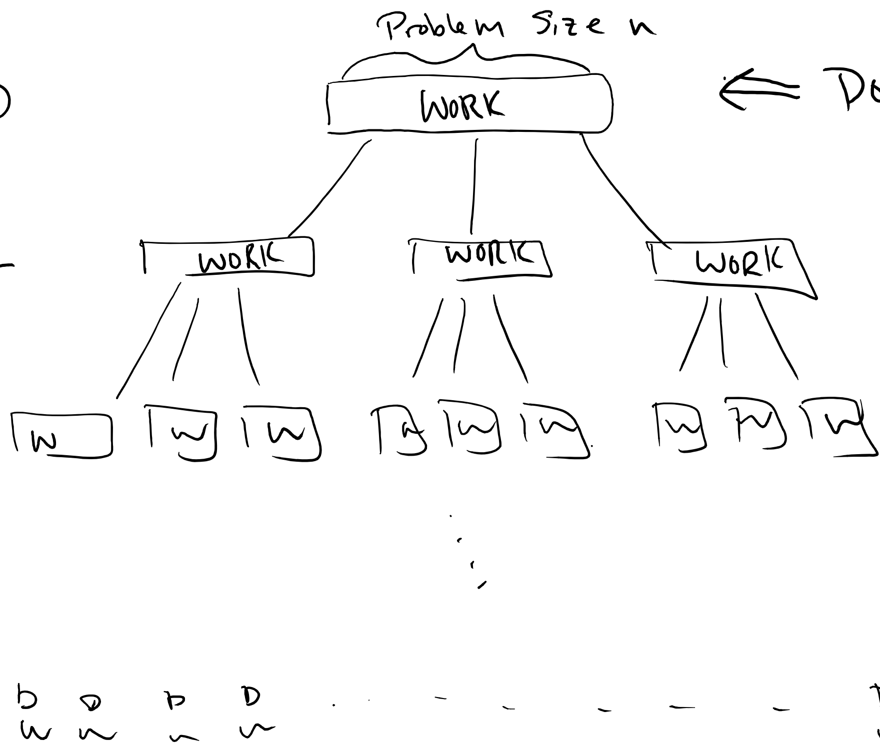
Gave you a rigorous method for figuring out work done in recursive algorithm: We added up all work

level 0

level 1

level 2

level F



$\Leftarrow$ Dominates when $a<b^d$

Dominates when $a>b^d$

$\nwarrow$ There are $n^{\log_b a}$ leaves!

$$T(n) = aT\left(\frac{n}{b}\right) + O(n^d)$$

Three Cases

$$T(n) = \begin{cases} O(n^d \log n) & a = b^d \\ O(n^d) & a < b^d \\ O(n^{\log_b a}) & a > b^d \end{cases}$$

$a = b^d$ $\Leftarrow$ Balanced

$a < b^d$ $\Leftarrow$ Work outside recursive call dominates

$a > b^d$ $\Leftarrow$ Work in recursive call dominates

S.KIMMEL

Loop Invariants: Prove loops are correct

```
setup
while (condition) {
    Stuff
}
Great output
```

Parts of Proof: (guess)

1. State invariant: this is thing that is true after every
2. Base Case: Show invariant is true before loop starts
3. Maintenance: Show if invariant is true before an iteration, it is also true after.
4. Termination: argue loop ends. State what invariant tells us given ending conditions

CS302 - Worksheet 1

Input : Array A of integers of length n
Output: Smallest value of A

$s = A[1]; i = 2;$
while $i \leq n$ **do**
    if $A[i] < s$ **then**
        $s = A[i]$
    end
    $i += 1;$
end
return s;

Algorithm 1: `Smallest`(A, n)

Solution We will prove the following loop invariants: (i) $s \in A[1 : i-1]$, (ii) $s \leq A[k]$ for k such that $1 \leq k \leq i-1$.

Base case: Before the loop starts, $i = 2$ and $s = A[1]$, so (i) and (ii) hold.

Maintenance: Let i_a, s_a be initial values, and i_b, s_b be new values at the end of the loop. Then

$$i_b = i_a + 1, \tag{1}$$
$$s_b = \min\{s_a, A[i_a]\}. \tag{2}$$

By assumption (ii), s_a is less than all values of $A[1 : i_a - 1]$, so combining this assumption with Eq. 2 and the fact that $A[i_a] = A[i_b - 1]$, (ii) is maintained. If $s_b = s_a$ then $s_b \in A[1 : i_a - 1] \subset A[1 : i_b - 1]$ by assumption (i). Otherwise, $s_b = A[i_a] = A[i_b - 1]$. Either way, (i) holds.

Termination: i increases with each loop, so the loop will terminate at $i = n$. Then we have s is the smallest value of the whole array by (i) and (ii).

S.KIMMEL

Q: Prove that $y=c$ at end of loop:

```
y=0; x=c;
while (x>0) {
    x--;
    y++;
}
```

Invariant: $x+y=c$

Base: $y=0$, $x=c$ ✓

Maintenace: $y_b = y_a + 1$, $x_b = x_a - 1$

$$x_b + y_b = y_a + 1 + x_a - 1 = \boxed{x_a + y_a = c}$$

By inductive assumption

Termination: x gets smaller at each step, so terminates when $x=0$. Since $x+y=c \Rightarrow y=c$