Introduction to Recursion Trees

Ian Ludden

Learning Objective

- Given a recursively defined function, find its closed form by drawing a recursion tree and adding up the work at all levels.

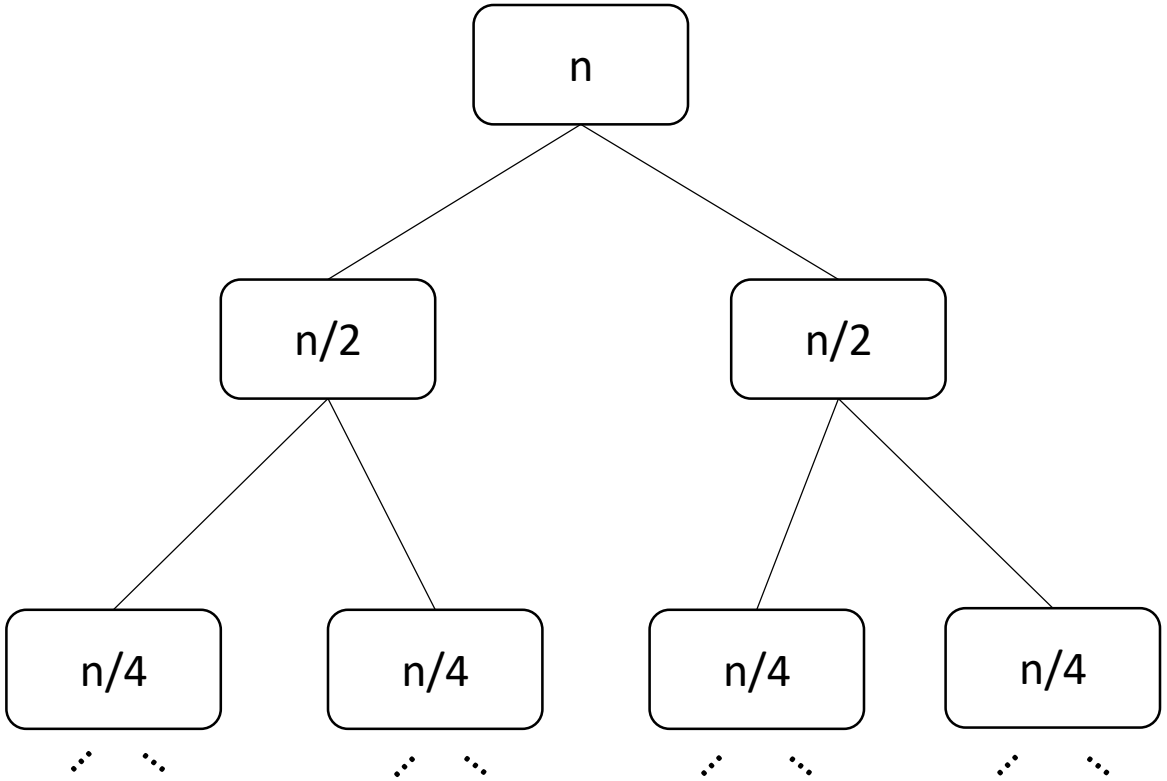
Recursion trees are visualization tools.

```
function helloUniverse(n):
  if n = 1 do
        print("Hello, world!")
        return
  endif

  for i from 1 to n do
    print("Hello, world!")
  endfor

  helloUniverse(floor(n/2))
  helloUniverse(floor(n/2))
endfunction
```

A silly recursive function

$$H(1) = 1$$

$$H(n) = n + 2 \cdot H\left(\left\lfloor \frac{n}{2} \right\rfloor\right) \ \forall n \geq 2$$



Example 1: Are You Really a Tree?

Matryoshka nesting dolls

Source: https://images.app.goo.gl/aFx8iAvh5hhRpGeg9

Recursive definition of total volume (in^3):

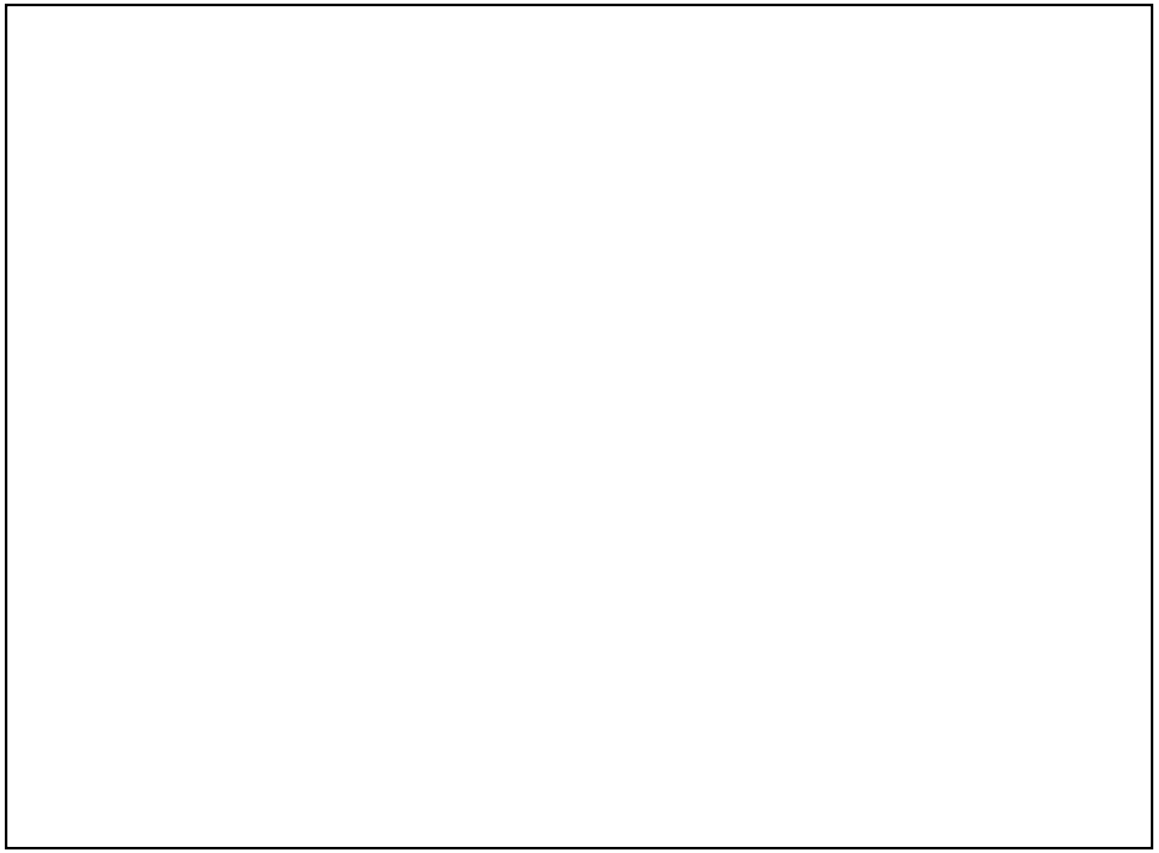
$V(1) = 1$

$V(n) = n + V(n-1)$

Level	Work
0	
1	
2	
3	
k	
h	

Example 2: The ~~Plot~~ Tree Thickens

$f(1) = 14; \quad f(n) = 2 \cdot f\left(\frac{n}{4}\right) + n^2 \quad \forall n \geq 2$ (assume n is a power of 4)



Level	Work Per Node	Total Level Work
0		
1		
2		
3		
k		
h		

Recap: Learning Objective

- Given a recursively defined function, find its closed form by drawing a recursion tree and adding up the work at all levels.