

Recurrences & mergesort analysis

Lecture 7

COSC 242 – Algorithms and Data Structures

Today's outline

1. Recurrence equations
2. Demonstration with a simpler equation
3. Solving mergesort with recurrence equations

Today's outline

1. Recurrence equations
2. Demonstration with a simpler equation
3. Solving mergesort with recurrence equations

Analysing divide-and-conquer

When an algorithm contains a recursive call to itself, we can often describe its running time by a **recurrence equation** or **recurrence**.

This equation describes the overall running time on a problem of size n in terms of the running time on smaller inputs

We call these **recurrence equations** because the function name T recurs on the righthand side of the equation.

Recurrence

$T(n)$ = running time on a problem of size n .

If the problem size is small enough (say, $n \leq c$ for some constant c), we have a base case. The brute-force solution takes constant time: $\Theta(1)$.

The base case is the small subproblem that is solved directly. They cause the recursion to terminate and “recurse” up to give the solution.

Otherwise, suppose that we divide into a subproblems, each $1/b$ the size of the original. (In merge sort, both $a = b = 2$).

$D(n)$ = The time to divide a size- n problem

Recurrence



We have a subproblems to solve, each of size n/b

Each subproblem takes $T(n/b)$ time to solve.

We therefore spend $a \cdot T(n/b)$ time solving all subproblems.

$C(n)$ = time to combine solutions.

Recurrence

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq c, \\ aT(n/b) + D(n) + C(n) & \text{otherwise} \end{cases}$$

Recurrence for mergesort

The time complexity function for mergesort is:

$$T(1) = 1$$

$$T(n) = 2T(n/2) + n$$

It would be easier to compare this function to our landmark functions if we could find a simple non-recurrent formula defining the function T .

Solving is not always possible, but for mergesort we can.

Some technicalities

We're going to keep things simple. If we were in a maths class, the main equation above would be rendered more precisely as:

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + n$$

this occurs when n is odd, and so we end up with sub-problems of size $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$.

By assuming that n is a power of 2 we can avoid these complications.

Solving recurrences - substitution



The substitution method has two steps:

1. Guess the form of the solution using substitution and iteration
2. Use induction to find the constants, and that the guess is true

We substitute the guessed solution for the function when applying the inductive hypothesis to smaller values; hence the name “substitution method.”

Today's outline

1. Recurrence equations
2. Demonstration with a simpler equation
3. Solving mergesort with recurrence equations

A simpler problem

Let's start with a simpler function:

Base case $f(1) = 2$

Thing to solve $f(n) = f(n - 1) + 3$

A simpler problem

Let's start with a simpler function:

Base case $f(1) = 2$

Thing to solve $f(n) = f(n - 1) + 3$

The problem is that the Right Hand Side (RHS) is defined in terms of the Left Hand Side (LHS).

To solve, we want to eliminate the f from the RHS of the equation.

To do this, I like to break up my workspace into two halves ^[1]...

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$$k = 1 \quad f(n) = f(n - 1) + 3$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

Lets expand this



$k = 1$

$$f(n) = f(n - 1) + 3$$

$$f(n - 1) =$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$k = 1$

$$f(n) = f(n - 1) + 3$$

$f(n - 1) =$ Lets take our $f(n)$, and since we wish to expand $f(n-1)$, we need to subtract 1 from every n .

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$$k = 1 \quad f(n) = f(n - 1) + 3$$

$f(n - 1) =$ That is, where ever there is an n in our original function, lets insert $n-1$ into that, in our workspace.

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$k = 1$

$$f(n) = f(n - 1) + 3$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \end{aligned}$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$k = 1$

$$f(n) = f(n - 1) + 3$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \end{aligned}$$

Lets now plug our workspace $f(n-1)$ expansion back into our solution side

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$k = 1$

$$\begin{aligned} f(n) &= f(n - 1) + 3 \\ &= [f(n - 2) + 3] + 3 \end{aligned}$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \end{aligned}$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$$\begin{aligned} k=1 \quad f(n) &= f(n - 1) + 3 \\ &= [f(n - 2) + 3] + 3 \\ k=2 \quad &= f(n - 2) + 6 \end{aligned}$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \end{aligned}$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$k = 1$

$$\begin{aligned} f(n) &= f(n - 1) + 3 \\ &= [f(n - 2) + 3] + 3 \end{aligned}$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \end{aligned}$$

$k = 2$

$$= f(n - 2) + 6$$

Lets solve this



Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$k = 1$

$$f(n) = f(n - 1) + 3$$
$$= [f(n - 2) + 3] + 3$$

$k = 2$

$$= f(n - 2) + 6$$

$$f(n - 1) = f((n - 1) - 1) + 3$$
$$= f(n - 2) + 3$$

$f(n - 2) =$ Where ever there is an n in our original function, we now insert $n-2$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$$\begin{aligned} k=1 \quad f(n) &= f(n - 1) + 3 \\ &= [f(n - 2) + 3] + 3 \\ k=2 \quad &= f(n - 2) + 6 \end{aligned}$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \\ f(n - 2) &= f((n - 2) - 1) + 3 \\ &= f(n - 3) + 3 \end{aligned}$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$k = 1$

$$f(n) = f(n - 1) + 3$$
$$= [f(n - 2) + 3] + 3$$

$k = 2$

$$= f(n - 2) + 6$$

$$f(n - 1) = f((n - 1) - 1) + 3$$
$$= f(n - 2) + 3$$

$$f(n - 2) = f((n - 2) - 1) + 3$$
$$= f(n - 3) + 3$$

Lets now plug our workspace $f(n-2)$ expansion back into our solution side

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$k = 1$

$$\begin{aligned} f(n) &= f(n - 1) + 3 \\ &= [f(n - 2) + 3] + 3 \end{aligned}$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \end{aligned}$$

$k = 2$

$$\begin{aligned} &= f(n - 2) + 6 \\ &= [f(n - 3) + 3] + 6 \\ &= f(n - 3) + 9 \end{aligned}$$

$$\begin{aligned} f(n - 2) &= f((n - 2) - 1) + 3 \\ &= f(n - 3) + 3 \end{aligned}$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$$\begin{aligned} k=1 \quad f(n) &= f(n - 1) + 3 \\ &= [f(n - 2) + 3] + 3 \\ k=2 \quad &= f(n - 2) + 6 \\ &= [f(n - 3) + 3] + 6 \\ k=3 \quad &= f(n - 3) + 9 \end{aligned}$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \\ f(n - 2) &= f((n - 2) - 1) + 3 \\ &= f(n - 3) + 3 \end{aligned}$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$$\begin{aligned} k=1 \quad f(n) &= f(n - 1) + 3 \\ &= [f(n - 2) + 3] + 3 \\ k=2 \quad &= f(n - 2) + 6 \\ &= [f(n - 3) + 3] + 6 \\ k=3 \quad &= f(n - 3) + 9 \end{aligned}$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \\ f(n - 2) &= f((n - 2) - 1) + 3 \\ &= f(n - 3) + 3 \\ f(n - 3) &= f((n - 3) - 1) + 3 \\ &= f(n - 4) + 3 \end{aligned}$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$$\begin{aligned} k=1 \quad f(n) &= f(n - 1) + 3 \\ &= [f(n - 2) + 3] + 3 \\ k=2 \quad &= f(n - 2) + 6 \\ &= [f(n - 3) + 3] + 6 \\ k=3 \quad &= f(n - 3) + 9 \\ &= [f(n - 4) + 3] + 9 \\ k=4 \quad &= f(n - 4) + 12 \end{aligned}$$

$$\begin{aligned} f(n - 1) &= f((n - 1) - 1) + 3 \\ &= f(n - 2) + 3 \\ f(n - 2) &= f((n - 2) - 1) + 3 \\ &= f(n - 3) + 3 \\ f(n - 3) &= f((n - 3) - 1) + 3 \\ &= f(n - 4) + 3 \end{aligned}$$

Solve the recurrence

Base case: $f(1) = 2$

Function: $f(n) = f(n - 1) + 3$

Solution

Workspace

$$\begin{aligned} k=1 \quad f(n) &= f(n-1) + 3 \\ &= [f(n-2) + 3] + 3 \end{aligned}$$

$$\begin{aligned} f(n-1) &= f((n-1)-1) + 3 \\ &= f(n-2) + 3 \end{aligned}$$

$$\begin{aligned} k=2 \quad &= f(n-2) + 6 \\ &= [f(n-3) + 3] + 6 \end{aligned}$$

$$\begin{aligned} f(n-2) &= f((n-2)-1) + 3 \\ &= f(n-3) + 3 \end{aligned}$$

$$\begin{aligned} k=3 \quad &= f(n-3) + 9 \\ &= [f(n-4) + 3] + 9 \end{aligned}$$

$$\begin{aligned} f(n-3) &= f((n-3)-1) + 3 \\ &= f(n-4) + 3 \end{aligned}$$

$$k=4 \quad = f(n-4) + 12$$

..

$$\mathbf{f(n) = f(n - k) + 3k}$$

Eliminating f on the right

How do we get rid of the f on the right?

Another way to think about it is: how far do we need to expand until the process stops? It stops at our base case. That is, $f(1) = 2$.

In other words, it stops when $n - k = 1$. That is, when $n - 1 = k$

$$f(n) = f(n - k) + 3k$$

Eliminating f on the right

How do we get rid of the f on the right?

Another way to think about it is: how far do we need to expand until the process stops? It stops at our base case. That is, $f(1) = 2$.

In other words, it stops when $n - k = 1$. That is, when $n - 1 = k$

$$\begin{aligned} f(n) &= f(n - k) + 3k \\ &= f(n - (n - 1)) + 3(n - 1) \end{aligned}$$

Eliminating f on the right

How do we get rid of the f on the right?

Another way to think about it is: how far do we need to expand until the process stops? It stops at our base case. That is, $f(1) = 2$.

In other words, it stops when $n - k = 1$. That is, when $n - 1 = k$

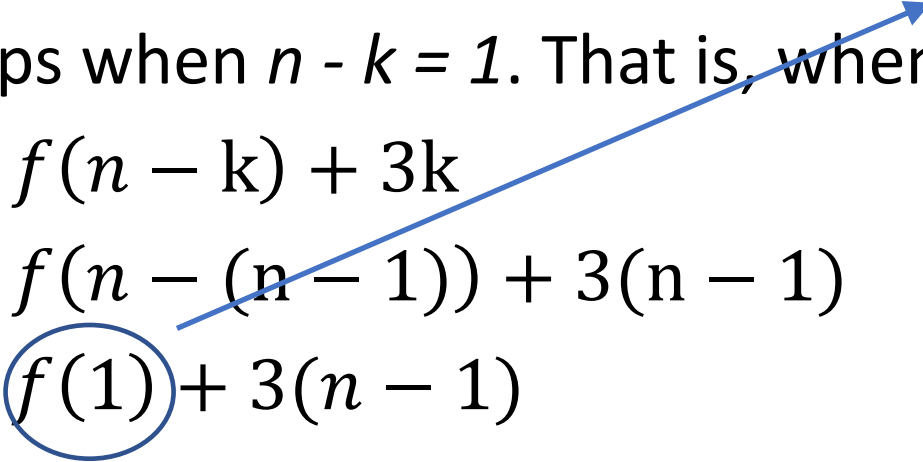
$$\begin{aligned} f(n) &= f(n - k) + 3k \\ &= f(n - (n - 1)) + 3(n - 1) \\ &= f(1) + 3(n - 1) \end{aligned}$$

Eliminating f on the right

How do we get rid of the f on the right?

Another way to think about it is: how far do we need to expand until the process stops? It stops at our base case. That is, $f(1) = 2$.

In other words, it stops when $n - k = 1$. That is, when $n - 1 = k$

$$\begin{aligned} f(n) &= f(n - k) + 3k \\ &= f(n - (n - 1)) + 3(n - 1) \\ &= f(1) + 3(n - 1) \end{aligned}$$


Here's our base case, which is = 2

Eliminating f on the right

How do we get rid of the f on the right?

Another way to think about it is: how far do we need to expand until the process stops? It stops at our base case. That is, $f(1) = 2$.

In other words, it stops when $n - k = 1$. That is, when $n - 1 = k$

$$\begin{aligned} f(n) &= f(n - k) + 3k \\ &= f(n - (n - 1)) + 3(n - 1) \\ &= f(1) + 3(n - 1) \\ &= 2 + 3(n - 1) \end{aligned}$$

Eliminating f on the right

How do we get rid of the f on the right?

Another way to think about it is: how far do we need to expand until the process stops? It stops at our base case. That is, $f(1) = 2$.

In other words, it stops when $n - k = 1$. That is, when $n - 1 = k$

$$\begin{aligned} f(n) &= f(n - k) + 3k \\ &= f(n - (n - 1)) + 3(n - 1) \\ &= f(1) + 3(n - 1) \\ &= 2 + 3(n - 1) \\ &= 3n - 1 \end{aligned}$$

Step 1 complete, we've guessed the non-recurrent form

Proving it

At this point we have a hypothesis which we need to prove:

Prove that the function f :

$$f(1) = 2$$

$$f(n) = f(n - 1) + 3$$

} The equation way back on Slide #12

And the function g :

$$g(n) = 3n - 1$$

are the same function.

$$f(1) = 2$$

$$f(n) = f(n - 1) + 3$$

$$g(n) = 3n - 1$$

$$n - 1 = k$$

Use induction

Base case: $f(1) = 2 = g(1)$

Inductive step: assume it's true for $n = k$. That is assume that $f(k) = g(k)$.
In other words assume that $f(k) = 3k - 1$.

Show that $f(k + 1) = g(k + 1)$.

$$\text{LHS} = f(k + 1)$$

Use induction

$$f(1) = 2$$

$$f(n) = f(n - 1) + 3$$

$$g(n) = 3n - 1$$

$$n - 1 = k$$

Base case: $f(1) = 2 = g(1)$

Inductive step: assume it's true for $n = k$. That is assume that $f(k) = g(k)$.
In other words assume that $f(k) = 3k - 1$.

Show that $f(k + 1) = g(k + 1)$.

$$\begin{aligned} \text{LHS} &= f(k + 1) \\ &= f(k) + 3 \end{aligned}$$

$$f(n) = f(n - 1) + 3$$

Lets replace 'n' with 'k+1'

$$f(n) = f(k + 1 - 1) + 3$$

$$f(1) = 2$$

$$f(n) = f(n - 1) + 3$$

$$g(n) = 3n - 1$$

Use induction


Base case: $f(1) = 2 = g(1)$

Inductive step: assume it's true for $n = k$. That is assume that $f(k) = g(k)$.
In other words assume that $f(k) = 3k - 1$.

Show that $f(k + 1) = g(k + 1)$.

$$\begin{aligned} \text{LHS} &= f(k + 1) \\ &= f(k) + 3 \\ &= 3k - 1 + 3 \end{aligned}$$

Substituting



$$f(1) = 2$$

$$f(n) = f(n - 1) + 3$$

$$g(n) = 3n - 1$$

Use induction

Base case: $f(1) = 2 = g(1)$

Inductive step: assume it's true for $n = k$. That is assume that $f(k) = g(k)$.
In other words assume that $f(k) = 3k - 1$.

Show that $f(k + 1) = g(k + 1)$.

$$\begin{aligned} \text{LHS} &= f(k + 1) \\ &= f(k) + 3 \\ &= 3k - 1 + 3 \\ &= 3(k + 1) - 1 \end{aligned}$$

Use induction

$$f(1) = 2$$

$$f(n) = f(n-1) + 3$$

$$g(n) = 3n - 1$$

Base case: $f(1) = 2 = g(1)$

Inductive step: assume it's true for $n = k$. That is assume that $f(k) = g(k)$.
In other words assume that $f(k) = 3k - 1$.

Show that $f(k+1) = g(k+1)$.

$$\begin{aligned} \text{LHS} &= f(k+1) \\ &= f(k) + 3 \\ &= 3k - 1 + 3 \\ &= 3(k+1) - 1 \end{aligned}$$

$$f(k+1) = g(k+1)$$

$$f(1) = 2$$

$$f(n) = f(n - 1) + 3$$

$$g(n) = 3n - 1$$

Use induction

Base case: $f(1) = 2 = g(1)$

Inductive step: assume it's true for $n = k$. That is assume that $f(k) = g(k)$.
In other words assume that $f(k) = 3k - 1$.

Show that $f(k + 1) = g(k + 1)$.

$$\begin{aligned} \text{LHS} &= f(k + 1) \\ &= f(k) + 3 \\ &= 3k - 1 + 3 \\ &= 3(k + 1) - 1 \end{aligned}$$

$$f(k+1) = g(k + 1)$$

$$\text{LHS} = \text{RHS}$$

Today's outline

1. Recurrence equations
2. Demonstration with a simpler equation
3. Solving mergesort with recurrence equations

Recurrence for mergesort

The time complexity function for mergesort is:

$$T(1) = 1$$

$$T(n) = 2T(n/2) + n$$

Lets now solve this recurrent form using the substitution method.

1. Guess the form of the solution using substitution and iteration
2. Use induction to find the constants, and that the guess is true

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$T(n) = 2T(n/2) + n$$

Lets expand this



$$T(n/2) =$$

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$T(n) = 2T(n/2) + n$$

$T(n/2) =$ Lets take our $T(n)$, and since we wish to expand $T(n/2)$, we need to divide every n by 2.

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

$k = 1$

$$T(n) = 2T(n/2) + n$$

Workspace

$T(n/2) =$ Where ever there is an n in our original function, lets insert $n/2$ into that, in our workspace.

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$T(n) = 2T(n/2) + n$$

$$\begin{aligned} T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \end{aligned}$$

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$T(n) = 2T(n/2) + n$$

$$\begin{aligned} T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \end{aligned}$$

Lets now plug our workspace expansion
back into our solution side for $T(n/2)$

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$\begin{aligned} T(n) &= 2T(n/2) + n \\ &= 2[2T(n/4) + n/2] + n \end{aligned}$$

$$\begin{aligned} T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \end{aligned}$$

$k = 2$

$$= 4T(n/4) + 2n$$

Lets expand this



Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$\begin{aligned} T(n) &= 2T(n/2) + n \\ &= 2[2T(n/4) + n/2] + n \end{aligned}$$

$k = 2$

$$= 4T(n/4) + 2n$$

$$\begin{aligned} T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \end{aligned}$$

$T(n/4) =$ Where ever there is an n in our original function, we now insert $n/4$

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$\begin{aligned} T(n) &= 2T(n/2) + n \\ &= 2[2T(n/4) + n/2] + n \end{aligned}$$

$$\begin{aligned} T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \end{aligned}$$

$k = 2$

$$= 4T(n/4) + 2n$$

$$\begin{aligned} T(n/4) &= 2T(n/2/4) + n/4 \\ &= 2T(n/8) + n/4 \end{aligned}$$



Lets now plug our workspace $T(n/4)$ expansion back into our solution side

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$\begin{aligned} T(n) &= 2T(n/2) + n \\ &= 2[2T(n/4) + n/2] + n \end{aligned}$$

$$\begin{aligned} T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \end{aligned}$$

$k = 2$

$$\begin{aligned} &= 4T(n/4) + 2n \\ &= 4[2T(n/8) + n/4] + 2n \end{aligned}$$

$$\begin{aligned} T(n/4) &= 2T(n/2/4) + n/4 \\ &= 2T(n/8) + n/4 \end{aligned}$$

$k = 3$

$$= 8T(n/8) + 3n$$

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$\begin{aligned} T(n) &= 2T(n/2) + n \\ &= 2[2T(n/4) + n/2] + n \end{aligned}$$

$$\begin{aligned} T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \end{aligned}$$


$k = 2$

$$\begin{aligned} &= 4T(n/4) + 2n \\ &= 4[2T(n/8) + n/4] + 2n \end{aligned}$$

$$\begin{aligned} T(n/4) &= 2T(n/2/4) + n/4 \\ &= 2T(n/8) + n/4 \end{aligned}$$

$k = 3$

$$= 8T(n/8) + 3n$$

Lets expand this 

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$

$$\begin{aligned} T(n) &= 2T(n/2) + n \\ &= 2[2T(n/4) + n/2] + n \end{aligned}$$

$$\begin{aligned} T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \end{aligned}$$

$k = 2$

$$\begin{aligned} &= 4T(n/4) + 2n \\ &= 4[2T(n/8) + n/4] + 2n \end{aligned}$$

$$\begin{aligned} T(n/4) &= 2T(n/2/4) + n/4 \\ &= 2T(n/8) + n/4 \end{aligned}$$

$k = 3$

$$= 8T(n/8) + 3n$$

$$\begin{aligned} T(n/8) &= 2T(n/2/8) + n/8 \\ &= 2T(n/16) + n/8 \end{aligned}$$

Lets now plug our workspace $T(n/8)$ expansion back into our solution side

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$k = 1$	$T(n) = 2T(n/2) + n$
	$= 2[2T(n/4) + n/2] + n$
$k = 2$	$= 4T(n/4) + 2n$
	$= 4[2T(n/8) + n/4] + 2n$
$k = 3$	$= 8T(n/8) + 3n$
	$= 8[2T(n/16) + n/8] + 3n$
$k = 4$	$= 16T(n/16) + 4n$

$T(n/2) = 2T(n/2/2) + n/2$
$= 2T(n/4) + n/2$
$T(n/4) = 2T(n/2/4) + n/4$
$= 2T(n/8) + n/4$
$T(n/8) = 2T(n/2/8) + n/8$
$= 2T(n/16) + n/8$

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$$\begin{aligned}k = 1 \quad T(n) &= 2T(n/2) + n \\ &= 2[2T(n/4) + n/2] + n \\k = 2 \quad &= 4T(n/4) + 2n \\ &= 4[2T(n/8) + n/4] + 2n \\k = 3 \quad &= 8T(n/8) + 3n \\ &= 8[2T(n/16) + n/8] + 3n \\k = 4 \quad &= 16T(n/16) + 4n\end{aligned}$$

$$\begin{aligned}T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \\T(n/4) &= 2T(n/2/4) + n/4 \\ &= 2T(n/8) + n/4 \\T(n/8) &= 2T(n/2/8) + n/8 \\ &= 2T(n/16) + n/8\end{aligned}$$

We want to get rid of the T on the right. We know that $T(1) = 1$, so what do we need to divide n by to get 1?

And what do we notice about the other constants (16 and 4 in the last equation)?

Solve the mergesort recurrence

Base case: $T(1) = 1$

Function: $T(n) = 2T(n/2) + n$

Solution

Workspace

$$\begin{aligned}k = 1 \quad T(n) &= 2T(n/2) + n \\ &= 2[2T(n/4) + n/2] + n \\k = 2 \quad &= 4T(n/4) + 2n \\ &= 4[2T(n/8) + n/4] + 2n \\k = 3 \quad &= 8T(n/8) + 3n \\ &= 8[2T(n/16) + n/8] + 3n \\k = 4 \quad &= 16T(n/16) + 4n \\ &= \mathbf{kT(n/k) + n \log k}\end{aligned}$$

$$\begin{aligned}T(n/2) &= 2T(n/2/2) + n/2 \\ &= 2T(n/4) + n/2 \\T(n/4) &= 2T(n/2/4) + n/4 \\ &= 2T(n/8) + n/4 \\T(n/8) &= 2T(n/2/8) + n/8 \\ &= 2T(n/16) + n/8\end{aligned}$$

We want to get rid of the T on the right. We know that $T(1) = 1$, so what do we need to divide n by to get 1?

And what do we notice about the other constants (16 and 4 in the last equation)?

Hypothesise a non-recurrent equation

We can now hypothesise that:

$$T(1) = 1$$

$$T(n) = 2T(n/2) + n$$

and

$$g(n) = n \log n + n$$


Base case $T(1) = 1$. This is when $n/k = 1$
Or, $n = k$. Substituting, when $n = k$, we get
 $n*1 + n*\log n$

are the same function for $n \geq 1$ and n is a power of two.

Note: we're going to ignore the in between steps when n is not a power of two. This means that our next element is not $k+1$ it is $2k$, because $n = \{1, 2, 4, 8, \dots, k, 2k, 4k, \dots\}$.

$$\begin{aligned}T(1) &= 1 \\T(n) &= 2T(n/2) + n \\g(n) &= n \log n + n\end{aligned}$$

Use induction

Base case: $T(1) = 1 = g(1)$ 

It stops at our base case. That is, $f(1) = 1$.
This time it's when $T(n/k) = T(1) = 1$.
That is, when $n/k = 1$, which is $n = k$

Inductive step: Assume that $T(k) = g(k)$. That is, assume $T(k) = k \log k + k$.

Show that $T(2k) = g(2k)$. That is, show $T(2k) = (2k) \log(2k) + 2k$

$$LHS = T(2k)$$

Use induction

$$\begin{array}{l} T(1) = 1 \\ T(n) = 2T(n/2) + n \\ g(n) = n \log n + n \end{array}$$

Base case: $T(1) = 1 = g(1)$

Inductive step: Assume that $T(k) = g(k)$. That is, assume $T(k) = k \log k + k$.

Show that $T(2k) = g(2k)$. That is, show $T(2k) = (2k) \log(2k) + 2k$

$$LHS = T(2k) = 2T(k) + 2k$$

Substitute $2k$ for n

Use induction

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 2T(n/2) + n \\ g(n) &= n \log n + n \end{aligned}$$

Base case: $T(1) = 1 = g(1)$

Inductive step: Assume that $T(k) = g(k)$. That is, assume $T(k) = k \log k + k$.

Show that $T(2k) = g(2k)$. That is, show $T(2k) = (2k) \log(2k) + 2k$

$$\begin{aligned} LHS &= T(2k) = 2T(k) + 2k \\ &= 2(k \log k + k) + 2k \end{aligned}$$

Using our induction hypothesis

$$\begin{aligned}T(1) &= 1 \\T(n) &= 2T(n/2) + n \\g(n) &= n \log n + n\end{aligned}$$

Use induction

Base case: $T(1) = 1 = g(1)$

Inductive step: Assume that $T(k) = g(k)$. That is, assume $T(k) = k \log k + k$.

Show that $T(2k) = g(2k)$. That is, show $T(2k) = (2k) \log(2k) + 2k$

$$\begin{aligned}LHS &= T(2k) = 2T(k) + 2k \\&= 2(k \log k + k) + 2k \\&= 2k(\log k + 1) + 2k\end{aligned}$$

$$\begin{aligned}T(1) &= 1 \\T(n) &= 2T(n/2) + n \\g(n) &= n \log n + n\end{aligned}$$

Use induction

Base case: $T(1) = 1 = g(1)$

Inductive step: Assume that $T(k) = g(k)$. That is, assume $T(k) = k \log k + k$.

Show that $T(2k) = g(2k)$. That is, show $T(2k) = (2k) \log(2k) + 2k$

$$\begin{aligned}LHS &= T(2k) = 2T(k) + 2k \\&= 2(k \log k + k) + 2k \\&= 2k(\log k + 1) + 2k \\&= 2k(\log k + \log 2) + 2k\end{aligned}$$

$$\begin{aligned}T(1) &= 1 \\T(n) &= 2T(n/2) + n \\g(n) &= n \log n + n\end{aligned}$$

Use induction

Base case: $T(1) = 1 = g(1)$

Inductive step: Assume that $T(k) = g(k)$. That is, assume $T(k) = k \log k + k$.

Show that $T(2k) = g(2k)$. That is, show $T(2k) = (2k) \log(2k) + 2k$

$$\begin{aligned}LHS &= T(2k) = 2T(k) + 2k \\&= 2(k \log k + k) + 2k \\&= 2k(\log k + 1) + 2k \\&= 2k(\log k + \log 2) + 2k \\&= 2k \log(2k) + 2k\end{aligned}$$

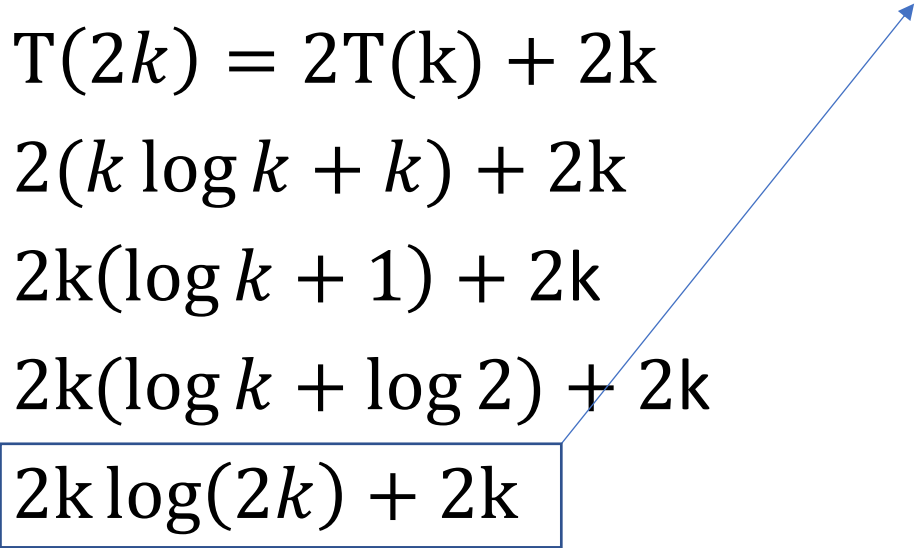
$$\begin{aligned}T(1) &= 1 \\T(n) &= 2T(n/2) + n \\g(n) &= n \log n + n\end{aligned}$$

Use induction

Base case: $T(1) = 1 = g(1)$

Inductive step: Assume that $T(k) = g(k)$. That is, assume $T(k) = k \log k + k$.

Show that $T(2k) = g(2k)$. That is, show $T(2k) = (2k) \log(2k) + 2k$

$$\begin{aligned}LHS &= T(2k) = 2T(k) + 2k \\&= 2(k \log k + k) + 2k \\&= 2k(\log k + 1) + 2k \\&= 2k(\log k + \log 2) + 2k \\&= 2k \log(2k) + 2k \\&= RHS\end{aligned}$$


Suggested reading

Chapter 4 looks at recurrences in some detail and is worth the read.

The substitution method is dealt with in Section 4.3, although we apply this method iteratively to develop the method further.

The recursion tree method is dealt with in Section 4.4. You may find this method helpful when trying to guess the form of the solution in the substitution method.

Section 4.5 and 4.6 look at the Master Theorem method which provides exact answers for many cases. But it is rather technical, not very interesting, and not needed for most cases.

References

1. Split workspace method courtesy of [Dr. John Bowers](#), at James Madison University.
2. You may find Dr Bowers' [walk-through video](#) on solving a recurrence equation with the substitution method helpful.

Image attributions

[This Photo](#) by Unknown Author is licensed under [CC BY](#)

Disclaimer: Images and attribution text provided by PowerPoint search. The author has no connection with, nor endorses, the attributed parties and/or websites listed above.