Solve the recurrence relation: $T(n) = \sqrt{n} T\left(\sqrt{n}\right) + n$

$$T(n) = \sqrt{n} T\left(\sqrt{n}\right) + n$$

Master method does not apply here. Recursion tree goes a long way. Iteration method would be preferable.

The answer is $\Theta(n \log \log n)$.

Can anyone arrive at the solution.

(recurrence-relations)

edited Nov 17 '12 at 18:44 Adi Dani 14.7k 3 17 39

asked Nov 17 '12 at 18:35 Vishnu Vivek 668 4 11 18

5 Answers

Let $n = m^{2^k}$. We then get that

$$T(m^{2^k}) = m^{2^{k-1}} T(m^{2^{k-1}}) + m^{2^k}$$

$$\begin{aligned} f_m(k) &= m^{2^{k-1}} f_m(k-1) + m^{2^k} = m^{2^{k-1}}\left(m^{2^{k-2}} f_m(k-2) + m^{2^{k-1}}\right) + m^{2^k} \\ &= 2m^{2^k} + m^{3 \cdot 2^{k-2}} f_m(k-2) \end{aligned}$$

$$m^{3\cdot 2^{k-2}} f_m(k-2) = m^{3\cdot 2^{k-2}}\left(m^{2^{k-3}} f_m(k-3) + m^{2^{k-2}}\right) = m^{2^k} + m^{7\cdot 2^{k-3}} f_m(k-3)$$

Hence,

$$f_m(k) = 2m^{2^k} + m^{3\cdot 2^{k-2}} f_m(k-2) = 3m^{2^k} + m^{7\cdot 2^{k-3}} f_m(k-3)$$

In general, it is not hard to see that

$$f_m(k) = \ell m^{2^k} + m^{(2^\ell - 1)2^{k-\ell}} f_m(k-\ell)$$

ℓ can go up to k, to give us

$$f_m(k) = km^{2^k} + m^{(2^k-1)} f_m(0) = km^{2^k} + m^{(2^k-1)} m^{2^0} = (k+1)m^{2^k}$$

This gives us

$$f_m(k) = (k+1)m^{2^k} = n\left(\log_2(\log_m n) + 1\right) = \mathcal{O}(n \log_2(\log_2 n))$$

since

$$n = m^{2^k} \implies \log_m(n) = 2^k \implies \log_2(\log_m(n)) = k$$

edited Nov 17 '12 at 19:11

answered Nov 17 '12 at 18:48 user17762

Thanks Marvis. But how did log appear suddenly. Are there some more steps in between. – Vishnu Vivek Nov 17 '12 at 18:56

@VISHNUVIVEK I have added some more details. Hope it is clear now. – user17762 Nov 17 '12 at 19:12

I appreciate your effort Marvis. I have a question for you. Is it possible to determine the base-case for a recurrence problem if it is not given in the question. Some say that it is not possible to solve recurrence problems if the base-case is not given. In this problem, we have to assume that the base case is T(2)=2. How did you solve it without knowing it. In the answer by Amr, he has left it at the end as T(2). Thus if we substitute T(2)=2, we almost arrive at the answer. – Vishnu Vivek Nov 17 '12 at 22:47

@VISHNUVIVEK The base case is $\mathcal{O}(1)$ and hence it typically won't affect the overall order. For instance, as Amr has shown that

$$T(m) = m\left(\log_2 \log_2(m) + \frac{T(2)}{2}\right)$$

$T(2)$ is just a constant say $T(2) = k$. Hence, the overall cost is

$$T(m) = m\left(\log_2 \log_2(m) + \frac{k}{2}\right) = \mathcal{O}(m \log_2 \log_2 m)$$

– user17762 Nov 17 '12 at 22:50

Oh! I almost forgot the fact that it was a constant term. Great. Thanks buddy. – Vishnu Vivek Nov 17 '12 at 22:58

at 22:58

Yes!!, the Master Theorem can be applied. I'm going to show you

T(n) = $\sqrt{n}$ T($\sqrt{n}$) + n = $\sqrt{n}$ T($\sqrt{n}$) + O(n)

Let n = 2^k, $\sqrt{n}$ = $2^{k/2}$, and k = $\log n$

T(2^k) = $2^{k/2}$T($2^{k/2}$) + 2^k / dividing by 2^k

$\frac{T(2^k)}{2^k} = \frac{2^{k/2}T(2^{k/2})}{2^k}$ + 1 (we know that $\frac{2^{k/2}}{2^k} = \frac{1}{2^{k/2}}$)

$\frac{T(2^k)}{2^k} = \frac{T(2^{k/2})}{2^{k/2}}$ + 1

Let y(k) = $\frac{T(2^k)}{2^k}$, then

y(k) = y($\frac{k}{2}$) + 1

Now, we apply the Master Theorem (T(n) = aT($\frac{n}{b}$) + n^d), where a=1, b=2, and d=0, a= b^d = 1

y(k) = $k^d \log k$ = $\log k$ (because d=0)

But, we also know that T(2^k) = $2^k y(k)$, then

T(2^k) = 2^k $\log k$ = T(n) = $n \log \log n$ (because n=2^k and k = $\log n$)

Finally: T(n) = $\Theta(n \log \log n)$

edited Sep 16 '14 at 19:46

answered Sep 16 '14 at 19:31
Pedro A. Rodriguez
51 2

Looks correct and generalizable to T(n) = sqrt(n)*T(sqrt(n)) + sqrt(n). – Eugene K Sep 24 '15 at 17:15

Let $n=2^{2^u}$, thus we get:

$$T(2^{2^u}) = 2^{2^{u-1}}T(2^{2^{u-1}}) + 2^{2^u}$$

Now divide both sides by 2^{2^u} to get (Note that $2^{2^{u-1}}/2^{2^u} = 2^{-2^{u-1}}$)

$$2^{-2^u}T(2^{2^u}) = 2^{-2^{u-1}}T(2^{2^{u-1}}) + 1$$

$$2^{-2^u}T(2^{2^u}) - 2^{-2^{u-1}}T(2^{2^{u-1}}) = 1$$

By summing from 1 to n we get:

$$2^{-2^n}T(2^{2^n}) - 2^{-1}T(2) = n$$

therefore:

$$T(2^{2^n}) = 2^{2^n}(2^{-1}T(2) + n)$$

answered Nov 17 '12 at 18:46
Amr
13.6k 3 24 79

Thanks Amr.. can u pls arrive till the answer which I've prescribed. – Vishnu Vivek Nov 17 '12 at 18:52

Actually I don't know the definitions of big O and theta (which you have). That's why I stopped here. I know they are used a lot in CS, but my field is not CS – Amr Nov 17 '12 at 18:53

The other answer has the big O notation. I think you might want to see this answer. – Amr Nov 17 '12 at 18:55

1 Actually your answer is quite understandable. I have a question for you. Is it possible to determine the base-case for a recurrence problem if it is not given in the question. – Vishnu Vivek Nov 17 '12 at 19:04

Solution with Detailed Explanation:

Master theorem **cannot be applied here** because for applying the Master theorem the number of sub-problems generated must be constant (a).

```
T(n)=√n*T(√n)+n
```

```
Let, m=lg n; n=2^m
```

Therefore

Therefore,

```
T(2^m)=2^(m/2)* T(2^(m/2)) + 2^m .....(i)

Let, S(m)=T(2^m)
```

Therefore changing the equation (i) we get,

```
S(m)=2^(m/2) * S(m/2) + 2^m .....(ii) //Level 1

S(m/2)=2^(m/4) * S(m/4) + 2^(m/2) .....(iii)

S(m/4)=2^(m/8) * S(m/8) + 2^(m/4) .....(iv)

....

....

So on and so forth.
```

By putting value of **S(m/4)** from equation (iv) in equation (iii) we get,

```
    S(m/2)=2^(m/4) * [2^(m/8) * S(m/8) + 2^(m/4)] + 2^(m/2)  //Replaced value is inside [
]

          =[(2^(m/4)*2^(m/8)) * S(m/8) + 2^(m/4) * 2^(m/4)] + 2^(m/2)

          =[(2^(m/4)*2^(m/8)) * S(m/8) + 2^(m/2)] + 2^(m/2) .....(v) //Level 2
```

By putting value of equation (v) in equation (ii) we get,

```
   S(m)=2^(m/2) * [[(2^(m/4)*2^(m/8)) * S(m/8) + 2^(m/2)] + 2^(m/2)] + 2^m

          =[(2^(m/2)*2^(m/4)*2^(m/8)) * S(m/8) + 2^(m/2) *2^(m/2)] + 2^(m/2) * 2^(m/2)] +
2^m

          =[(2^(m/2)*2^(m/4)*2^(m/8)) * S(m/8) + 2^m] + 2^m] + 2^m  ......(vi) //Level 3
```

So, if you follow the expressions you will find in each level a factor of **(2^m)** is added and so **(2^(m/(2^i)))** is multiplied to **S(m/(2^i))**.

Note: You can verify it by comparing equation (ii) and equation (v) or equation (v) and equation (vi).

So it turns out,

```
S(m)=[(2^(m/(2^1))*2^(m/(2^2))*2^(m/(2^3))*....*2^(m/(2^i)))* S(m/(2^i))]

        +[2^m + 2^m + ..... i terms]
```

Note: You may directly reach to the above relation just after equation (ii) or (iii) depending on your expertise.

Now, if you follow the first part in [ ] you can make it,

```
2^((m/2)[1+(1/2)+(1/4)+...+(1/(2^(i-1))])  .....(vii)
```

The part in [ ] in the above line is a GP series with first term **a=1** and **r=1/2** with **i** terms. As this is a monotonically increasing series, if we make it for infinite terms, still it holds the upper bound.

Therefore, the sum of the series ie. the section inside [ ] has become,

```
1/(1-(1/2)) =2
```

So now putting the value in equation (vii), we get,

```
2^((m/2)*2) =2^m
```

Therefore,

```
S(m) <= 2^m * S(m/(2^i) + i * 2^m  //As we took the series sum for infinite terms
```

No if we consider **S(1)** as the base case and **S(1)=1**, then

```
m/(2^i) =1; and i=lg m; ..............(viii)
```

put the value of **i** in the equation (viii) to get

```
S(m) <= 2^m * S(1) + lg m * 2^m

     <= 2^m + lg m * 2^m     //As, S(1)=1
```

No return to **T(n)** by replacing the variables.

```
S(lg n) <= n + lg (lg n) * n

T(n) <= n + n * lg (lg n)
```

Therefore,

```
T(n) = O(n lg(lg n))
```

Now, for the approximation we made for the GP series in equation (vii) is not relevant for calculation that lower bound as calculation that lower bound as,

```
n = O(n lg(lg n))
```

So, we can conclude,

```
T(n) = θ(n lg(lg n)) //"Theta of n lg of lg of n"
```

answered Mar 3 '15 at 12:38

Partha De
11 1

Use substitution method:

$$\begin{aligned}
\mathrm{T}(n) &= \sqrt{n}\ \mathrm{T}(\sqrt{n}) + n \\
&= n^{\frac{1}{2}}\ \mathrm{T}\left(n^{\frac{1}{2}}\right) + n \\
&= n^{\frac{1}{2}}\left(n^{\frac{1}{2^2}}\ \mathrm{T}\left(n^{\frac{1}{2^2}}\right) + n^{\frac{1}{2}}\right) + n \\
&= n^{\frac{1}{2}+\frac{1}{2^2}}\ \mathrm{T}\left(n^{\frac{1}{2^2}}\right) + n^{\frac{1}{2}+\frac{1}{2}} + n \\
&= n^{\frac{1}{2}+\frac{1}{2^2}}\ \mathrm{T}\left(n^{\frac{1}{2^2}}\right) + 2n \\
&= n^{\frac{1}{2}+\frac{1}{2^2}}\left(n^{\frac{1}{2^3}}\ \mathrm{T}\left(n^{\frac{1}{2^3}}\right) + n^{\frac{1}{2^2}}\right) + 2n \\
&= n^{\frac{1}{2}+\frac{1}{2^2}+\frac{1}{2^3}}\ \mathrm{T}\left(n^{\frac{1}{2^3}}\right) + n^{\frac{1}{2}+\frac{1}{2^2}+\frac{1}{2^2}} + 2n \\
&= n^{\frac{1}{2}+\frac{1}{2^2}+\frac{1}{2^3}}\ \mathrm{T}\left(n^{\frac{1}{2^3}}\right) + 3n \\
&\vdots \\
&= n^{\sum_{i=1}^{k}\frac{1}{2^i}}\ \mathrm{T}\left(n^{\frac{1}{2^k}}\right) + kn
\end{aligned}$$

assuming $\mathrm{T}(2) = 2$, which is the least value of n that could be.
So,

$$\begin{aligned}
n^{\frac{1}{2^k}} &= 2 \\
\frac{1}{2^k}\log_2(n) &= \log_2(2) \\
\log_2(n) &= 2^k \\
\log_2\log_2(n) &= k\log_2(2) \\
\log_2\log_2(n) &= k
\end{aligned}$$

therefore, the recurrence relation will look like:

$$\begin{aligned}
\mathrm{T}(n) &= n^{\sum_{i=1}^{k}\frac{1}{2^i}}\ \mathrm{T}\left(n^{\frac{1}{2^k}}\right) + kn \\
&= n^{\sum_{i=1}^{\log_2\log_2(n)}\frac{1}{2^i}}\ \mathrm{T}\left(n^{\frac{1}{2^{\log_2\log_2(n)}}}\right) + n\log_2\log_2(n)
\end{aligned}$$

where,

$$\sum_{i=1}^{\log_2\log_2(n)}\frac{1}{2^i} = 1 - \frac{1}{\log_2(n)} = \text{fraction always, as } n \geq 2$$

so,

$$\mathrm{T}(n) = \mathcal{O}\left(n\log_2\log_2 n\right)$$

answered Oct 2 '16 at 9:46

amarVashishth
219 1 12