CS5800 – ALGORITHMS

MODULE 3. DIVIDE & CONQUER - I

Lesson 1: Recurrences

Ravi Sundaram

Topics

- Divide & Conquer – What is it
- Binary Search – example of Divide & Conquer
- Recurrences
- Guess & Check
- Recursion Tree
- Summary

Divide & Conquer – What is it?

- Political advice often attributed to Alexander the Great's father Philip, but certainly older
- The idea: crush the resistance by dividing it up - ensure they can't ally with each other against you
- In D&C algorithms, the data parts are separated and can't "talk to each other" until after the subproblems are solved
- But you've eventually got to get everything under control, provide the "glue" that keeps the empire together - that's often the hard part in algorithms
- Alexander tried to provide "cultural" glue for his conquests, but the empire still fell apart after he died

Divide & Conquer – What is it?

- To divide and conquer, think about how you could:
- Break the problem into recursive subproblems
- Combine the subproblem answers to create the solution to the whole problem.
- As with recursion in general, it helps during design to simply imagine what you would do if your algorithm worked for smaller instances.
- If base case and “inductive” case both work, the whole thing works.

Binary Search – example of Divide & Conquer

- Searching for element in ordered array

```
BinarySearch(low, high)
If (high < low) Then Return(not-found)
mid = (low + high) / 2
If (A[mid] > x) Then Return(BinarySearch(low, mid-1))
ElseIf (A[mid] < x) Then Return(BinarySearch(mid+1,high))
Else Return(mid)
```

- Divide step – break range (low, high) into (low, mid -1) and (mid + 1, high)
- Conquer step - combine subproblems by realizing one range has no solution

Recurrences

- Since the Divide & Conquer approach assumes solution to subproblems it naturally gives rise to recurrences.
- Binary search gives rise to the recurrence: $T(n) = T(n/2) + 1; T(1) = 1$
- Recurrences can be solved by a) Guess & Check, b) Recursion Tree method and c) Master Theorem.
- Guess and Check for Binary Search:

  guess $T(n) = \lg n$

  check $T(n/2) + 1 = \lg(n/2) + 1 = \lg n - 1 + 1 = \lg n$.
- Requires rigorous proof by induction

Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

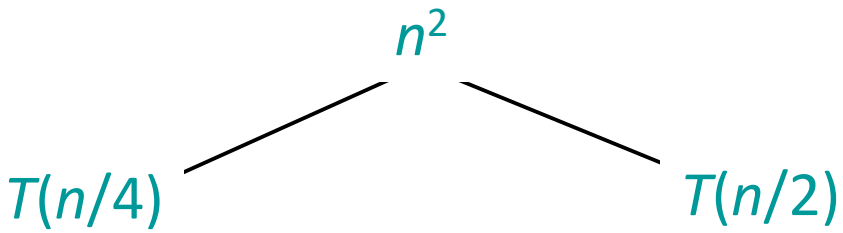
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

$T(n)$

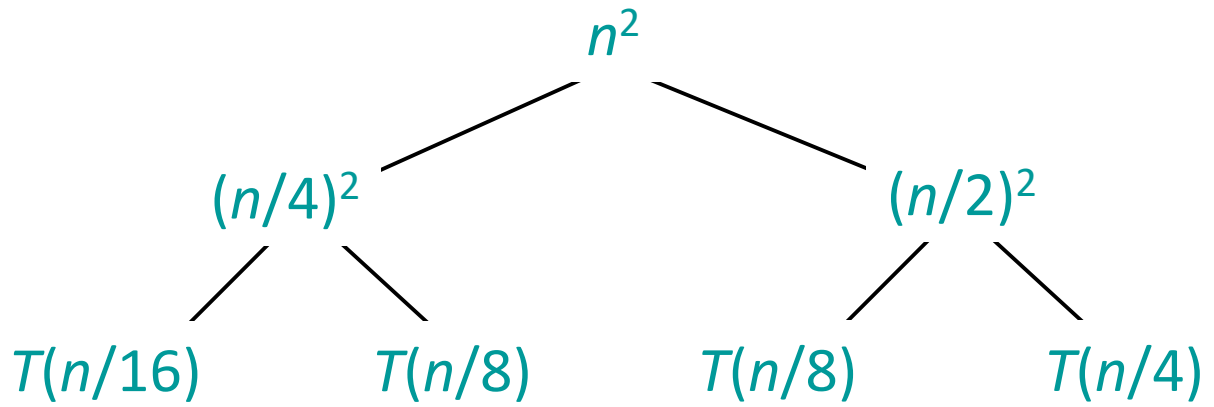
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



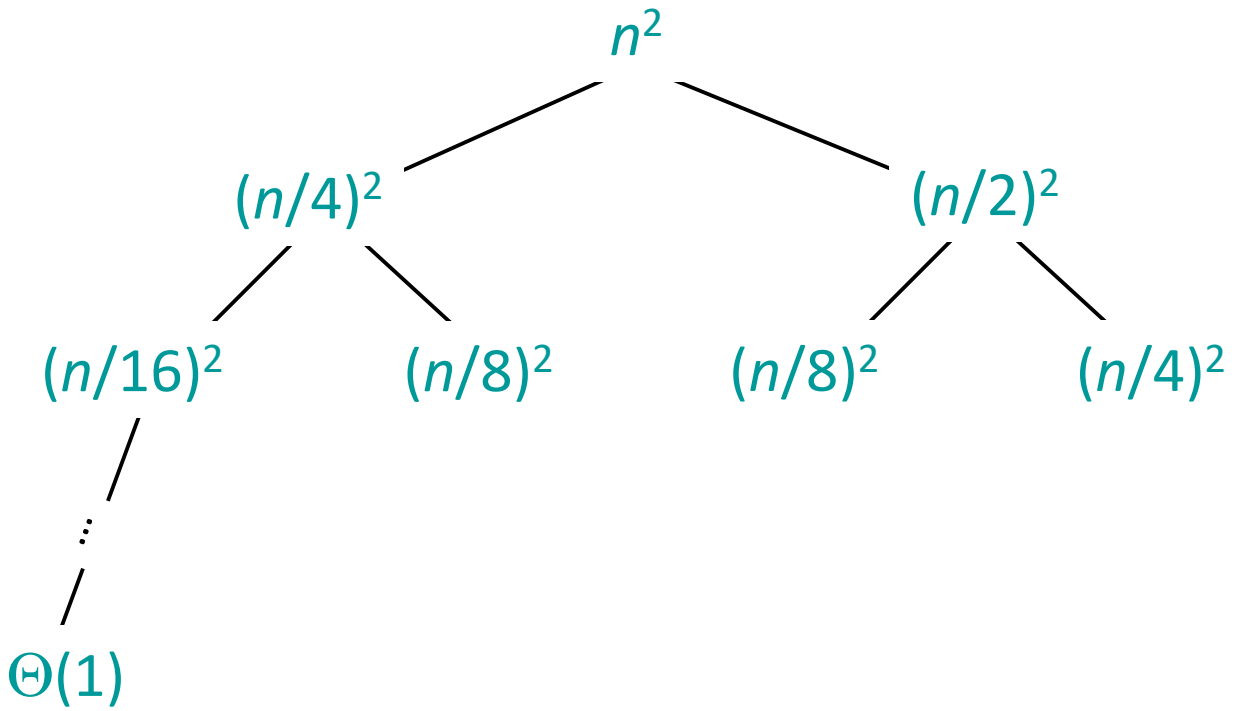
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



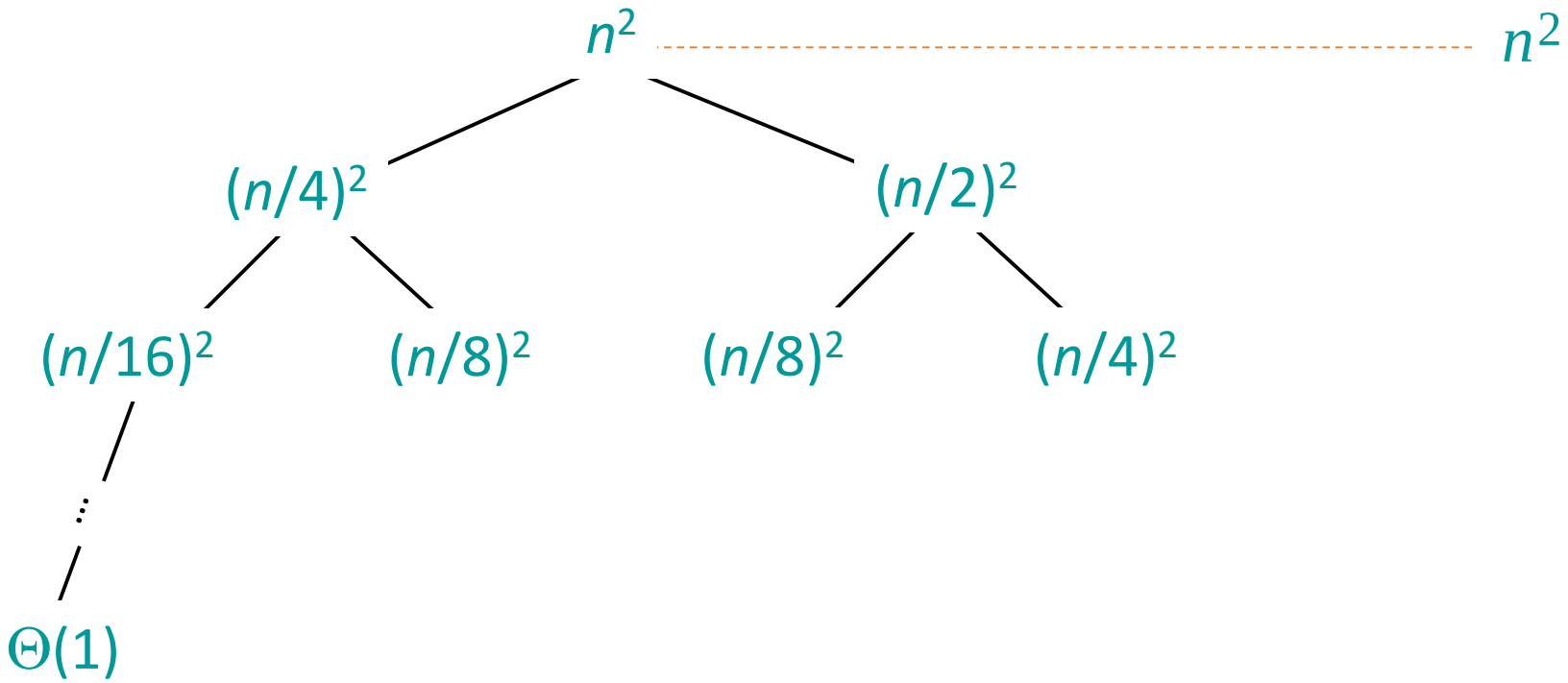
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



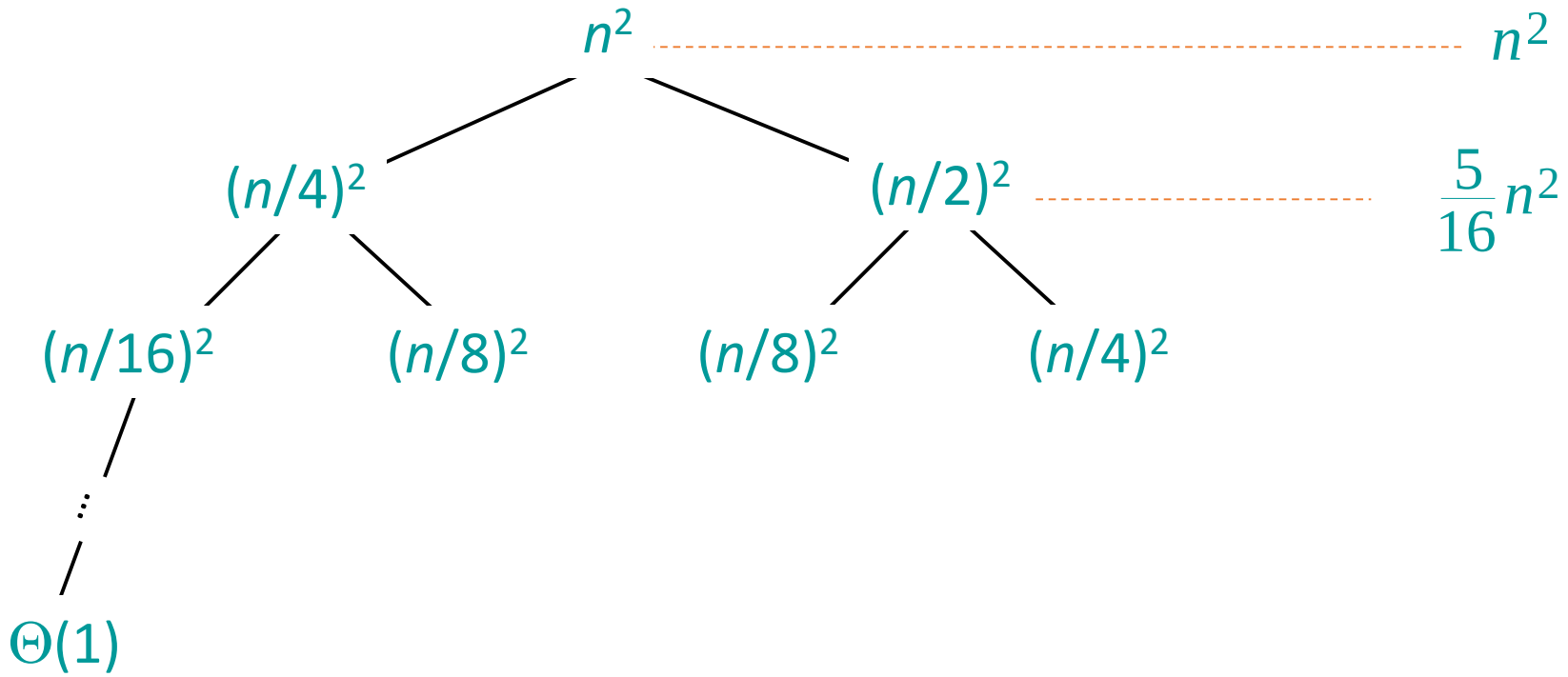
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



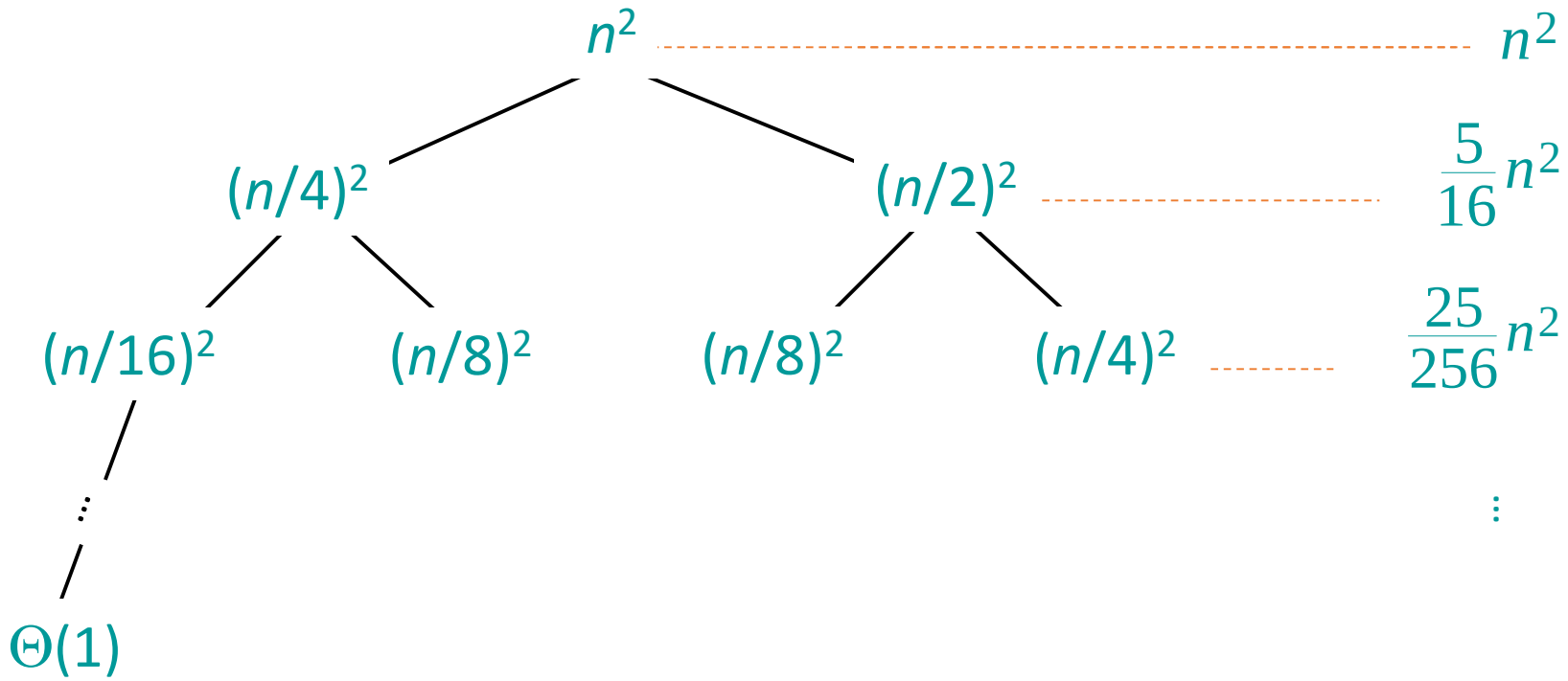
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



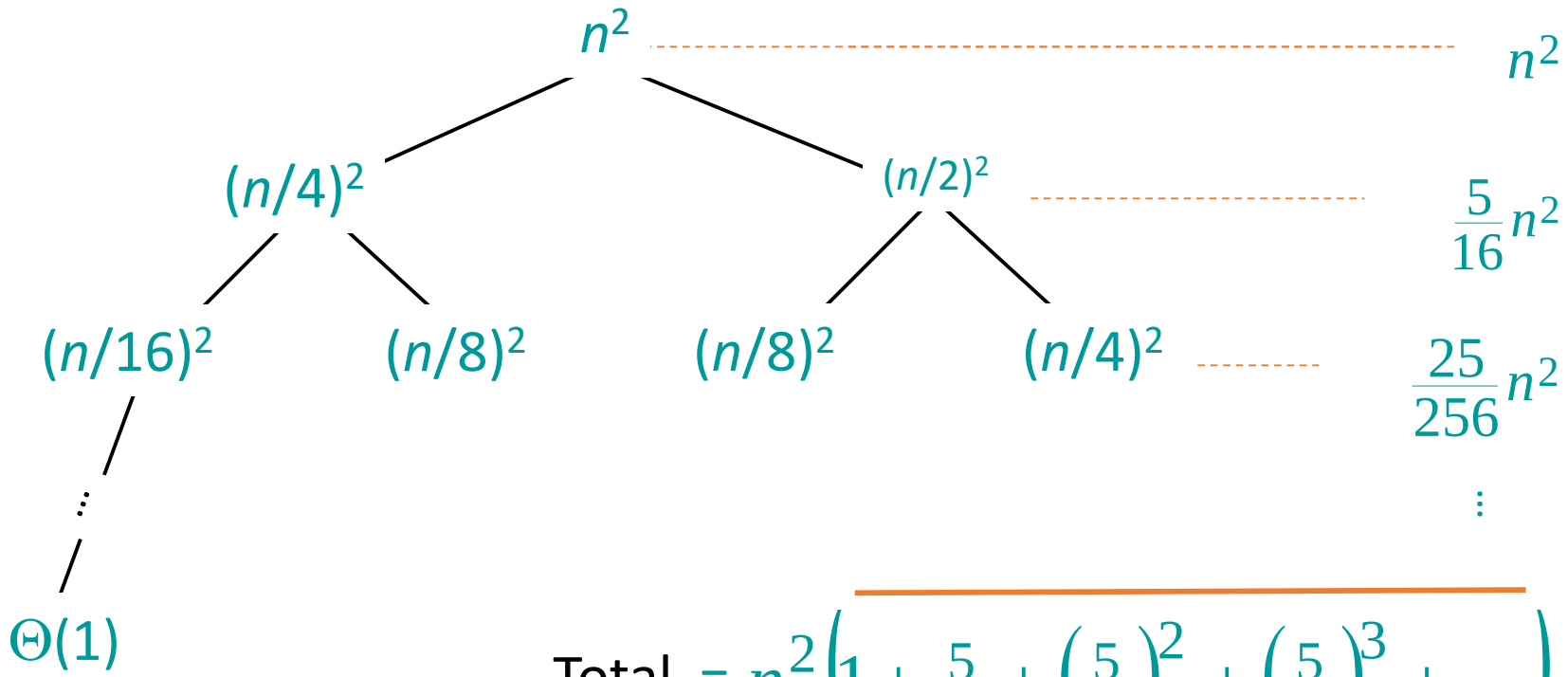
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



$$\text{Total} = n^2\left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots\right)$$

geometric series

$$= \Theta(n^2)$$

Summary

- Divide & Conquer is a recursive approach that assumes we can solve subproblems then asks how to combine the answers
- Because it assumes the solution for subproblems the analysis of the time complexity of Divide & Conquer algorithms naturally gives rise to recurrences
- Main takeaway: Recurrences can be solved using:
  - Guess & Check
  - Recursion tree
  - Master Theorem