Review

- Solving recursions
  - Repeated substitution
    - Works for recurrences of the form
      - \( T(n) = a \, T(n-1) + b \)
    - Linear homogeneous with constant coefficients
      - Example: Fibonacci recurrence \( F(n) = F(n-1) + F(n-2) \)
      - Recurrences of the form
        - \( T(n) = c_1 \, T(n-1) + c_2 \, T(n-2) + \ldots + c_r \, T(n-r) \)
Today’s Topic

• Solving recursions
  – Master Method
    • Works for recurrences of the form
      – \( T(n) = a \, T(n / b) + f(n) \)

Mergesort

• To sort a list of numbers \( L \)
  – Break up \( L \) into two lists of equal size: \( L1 \) & \( L2 \)
  – Recursively sort each list \( L1 \) & \( L2 \)
  – Merge together \( L1 \) & \( L1 \) (how?)
Mergesort

- To sort a list of numbers L
  
  ```
  Mergesort(L)
  If |L| = 1, Return L
  Else
    (L1, L2) = Split (L)
    Mergesort(L1)
    Mergesort(L2)
    L = Merge(L1, L2)
    Return L
  ```

- \( T(n) = 2 \ T(n/2) + n \)
  
  ```
  Mergesort(L)
  If |L| = 1, Return L
  Else
    (L1, L2) = Split (L)
    Mergesort(L1)
    Mergesort(L2)
    L = Merge(L1, L2)
    Return L
  ```
Recurrence form

- Recursion of the form
  - \( T(n) = a \cdot T(n / b) + f(n) \)
  - Mergesort: \( 2 \cdot T(n/2) + n \)
    - \( a = 2 \)
    - \( b = 2 \)
    - \( f(n) = n \)

- \( T(n) = 3 \cdot T(n / 2) + c \cdot n^2 \) ?

Recurrence Trees

- \( T(n) = 3 \cdot T(n / 2) + c \cdot n^2 \)
Recurrence Trees

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

By Term Expansion

- $T(n) = 4T(n/2) + n$
More Examples

- \( T(n) = 2T(n / 4) + \sqrt{n} \)

Recurrence form

• Why do we care?
  - \( T(n) = a \, T(n / b) + f(n) \)
Divide-and-Conquer

• Solving a problem of $T(n)$ by breaking it into smaller parts
  – $T(n) = a \cdot T(n / b) + f(n)$
  – benefit?
  – Examples:
    • Mergesort
    • Shortest distance
    • Binary search

Back to the tree: Master Method

• Given
  – $T(n) = a \cdot T(n / b) + f(n)$
• How do $a$, $b$, $f(n)$ determine the shape of the tree?
• Guess $T(n)$ depends on what?
Limited Master Method

- \( T(n) = a \cdot T(n / b) + n \)

Evaluate:
- Case 1: \( a/b < 1 \)
- Case 2: \( a/b > 1 \)
- Case 3: \( a/b = 1 \)
Back to the analysis

• Given
  – $T(n) = a \ T(n \ / \ b) + n$

• How do $a$, $b$, $f(n)$ determine the shape of the tree?
• Guess $T(n)$ depends on what?

Examples of Limited Master Method

– $T(n) = 3 \ T(n/2) + n$

– $T(n) = 4 \ T(n/4) + n$

– $T(n) = 4 \ T(n/5) + n$
Master Method

- $T(n) = a \cdot T(n/b) + f(n)$

**Case 1:** $f(n)$ “larger” than $n^{\log_b a}$

Master Method

- $T(n) = a \cdot T(n/b) + f(n)$

**Case 1:** $f(n)$ “smaller” than $n^{\log_b a}$
Master Method

- \( T(n) = a \ T(n / b) + f(n) \)

\[ \begin{array}{c}
\text{f(n)} \\
\downarrow \\
\text{f(n/b)} \\
\downarrow \\
\text{f(n/b^2)} \\
\downarrow \\
\vdots \\
\downarrow \\
\text{f(n/b^2)} \\
\downarrow \\
\text{f(n/b^2)} \\
\downarrow \\
\ddots \\
\end{array} \]

Case 1: \( f(n) \) “equal” \( n^{\log_b a} \)

Master Method Summary

- Given:
  - \( T(n) = a \ T(n / b) + f(n) \)

- 3 general cases: (in the long run)
  - \( f(n) \) grows faster than \( n^{\log_b a} \)
  - \( f(n) \) grows slower than \( n^{\log_b a} \)
  - \( f(n) \) grows proportional to \( n^{\log_b a} \)
Master Method Notes

• “smaller”, “larger” meaning?
  - must be some factor of $n^c$ different

• “Equal” means within a constant factor

• $f(n)$ may be in none of the three cases!

Examples of Limited Master Method

- $T(n) = 3 \cdot T(n/2) + n^2$

- $T(n) = 3T(n/2) + n^{1.4}$

- $T(n) = 16 \cdot T(n/4) + 10n^2 + 3n + 5$
Review on Recursion

• A problem containing sub-problems of the same structure
  – \( T(n) = T(sth < n) \ldots \)
  – Initial condition \( T(0), T(1), \ldots \)

• Sequence

• Algorithm

Review on Recursion

• Recursive representation: typically simple & beautiful
• What’s the real structure under the recursion coat?
  – Sequence

  – Algorithm