Recurrence Relations
- model lots of problems
  • The Tower of Hanoi
  • Divide and conquer algorithms
    – Sorting algorithm mergesort
    – Sorting algorithm quicksort
  • Tree algorithms
    – Searching for an element in a binary search tree
    – Listing out all elements in a binary search tree

Solving a recurrence relation
  • Problem sets up as a recurrence
    – Must have a base case
  • Solve the recurrence
    – Use substitution
  • Prove correctness
    – Proof by induction
Example 1

• $a_n = a_{n-1} + c$
• $a_0 = 1$

• Solve recurrence
• Then prove true by induction
• What is this an example of?

Worst case binary search tree

Example 2

• $a_n = 2*a_{n-1} + c$
• $a_0 = 0$

• Solve recurrence
• Then prove true by induction
• What is this an example of?

Towers of Hanoi

• Figures
  – Figs 1-4
    • problem size n-1
  – Figs 4-5
    • Constant work
  – Figs 5-7
    • problem size n-1
Example 3

- \( a_n = 2a_{n/2} + c \)
- \( a_1 = c \)

Solve recurrence
Then prove true by induction
What is this an example of?

Traversal in binary search tree
preorder, postorder, inorder

Example 4

- \( a_n = 2a_{n/2} + cn \)
- \( a_1 = c \)

Solve recurrence
Then prove true by induction
What is this an example of?

MergeSort

- \( n \log n \)
Definition

- A linear homogeneous recurrence relation of degree k with constant coefficients is a recurrence relation of the form
- \[ a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k} \]
- Where \( c_i \) are real numbers and \( c_k \neq 0 \)

Theorem

- Let \( c_1 \) and \( c_2 \) be real numbers. Suppose that \( r^2 - c_1 r - c_2 = 0 \) has two distinct roots \( r_1 \) and \( r_2 \). Then the sequence \( \{a_n\} \) is a solution of the recurrence
  \[ a_n = c_1 a_{n-1} + c_2 a_{n-2} \]
  if and only if
  \[ a_n = \alpha_1 r_1^n + \alpha_2 r_2^n \]
  for all \( n \) where \( \alpha_1 \) and \( \alpha_2 \) are constants

Example

- What is the solution to the recurrence relation \( a_n = a_{n-1} + 2a_{n-2} \) with \( a_0 = 2 \) and \( a_1 = 7 \)

Many other theorems

- See theorems 2-6 in Chapter 8.2
Theorem 1 in 8.3

Let \( f \) be an increasing function that satisfies the recurrence relation

\[
f(n) = a f(n/b) + c
\]

whenever \( n \) is divisible by \( b \), where \( a \geq 1 \), \( b \) is an integer greater than 1, and \( c \) is a positive real number. Then

\[
f(n) = \begin{cases} 
O(n^{\log_b a}) & \text{if } a > 1, \\
O(\log n) & \text{if } a = 1.
\end{cases}
\]

Furthermore, when \( n = b^k \) and \( a \neq 1 \), where \( k \) is a positive integer,

\[
f(n) = C_1 n^{\log_b a} + C_2,
\]

where \( C_1 = f(1) + c/(a - 1) \) and \( C_2 = -c/(a - 1) \).

Master Theorem in 8.3

MASTER THEOREM Let \( f \) be an increasing function that satisfies the recurrence relation

\[
f(n) = a f(n/b) + cn^d
\]

whenever \( n = b^k \), where \( k \) is a positive integer, \( a \geq 1 \), \( b \) is an integer greater than 1, and \( c \) and \( d \) are real numbers with \( c \) positive and \( d \) nonnegative. Then

\[
f(n) = \begin{cases} 
O(n^d) & \text{if } a < b^d, \\
O(n^d \log n) & \text{if } a = b^d, \\
O(n^{\log_b a}) & \text{if } a > b^d.
\end{cases}
\]