

Discrete Mathematical Structures  
CS 3233 Lecture 16

Prof. William Winsborough  
October 24, 2006

## Business

- Read section 4.1 and 4.2 for Thursday
- Homework 7  
Due Tuesday 31 October
  - 3.3: 2, 14
  - 4.1: 4, 6
  - 4.2: 4
- Reminder: Winsborough will be out of town 10/30-11/2
  - Lecture will be covered by Prof. Tom Bylander
  - Recitation will be covered by Catherine

24 October 2006

Winsborough CS 3233 Lecture 16

2

## Computational Complexity of Algorithms

- This is where Big-O, Big-Omega, and Big-Theta get used
- Complexity is measured principally in terms of two resources
  - Time Complexity
  - Space Complexity
    - Discussed more in course on data structures
- Worst-case complexity vs. Average-case

24 October 2006

Winsborough CS 3233 Lecture 16

3

## Why is Big-O Handy?

- You don't have to worry about a lot of details

```
proc bubble sort(a1, a2, ..., an)
for i := 1 to n-1
  for j := 1 to n-i
    if aj > aj+1 then interchange aj and aj+1
```
- In pass i, n-i comparisons are performed
  - With out using Big-O, we have cost of
$$\sum_{1 \leq i \leq n} (n-i) = \sum_{0 \leq k \leq n-1} k = (n-1)n/2$$
    - Labor-intensive because of manipulating summations
  - Using Big-O, constant terms and factors are not significant
- In each of O(n) passes, O(n) comparisons are performed
  - Total cost: O(n<sup>2</sup>)

24 October 2006

Winsborough CS 3233 Lecture 16

4

Complexity	Terminology	Example
$O(1)$	Constant complexity	Find max value in first 100 elements
$O(\log n)$	Logarithmic complexity	Binary search
$O(n)$	Linear	Linear search
$O(n \log n)$	$n \log n$	Merge sort
$O(n^b)$	Polynomial	Bubble sort
$O(b^n), b > 1$	Exponential	Evaluate logical formulas via truth tables
$O(n!)$	Factorial	Traveling salesman

\* Considered *tractable*

24 October 2006 Winsborough CS 3233 Lecture 16 5

## Problems versus Algorithms

- The complexity of a problem is characterized by the complexity of the best algorithm for solving it
  - It is always possible to give a sub-optimal algorithm
- A significant part of computer science is concerned with finding optimal algorithms for important problems

24 October 2006 Winsborough CS 3233 Lecture 16 6

## Theory versus Practice

- Some algorithms that are intractable in the worst case usually execute much faster than algorithms that are always tractable
- If the intractable cases are rare enough, such algorithms may be preferable
- Thus, while Big-O analysis is helpful, it cannot be considered 100% conclusive in all cases about the relative merits of different algorithms

24 October 2006 Winsborough CS 3233 Lecture 16 7

## Mathematical Induction

- How can we show that a proposition  $P(n)$  holds for all natural numbers  $n \in \mathbb{N}$ ?
- Proof technique called *mathematical induction*:
  - Basis: show  $P(0)$
  - Inductive Step: show that for all  $k \in \mathbb{N}$ ,  $P(k) \rightarrow P(k+1)$
- The proposition  $P(k)$  is called the *induction hypothesis*
- $(P(0) \wedge \forall k(P(k) \rightarrow P(k+1))) \rightarrow \forall nP(n)$

24 October 2006 Winsborough CS 3233 Lecture 16 8

## Example of Induction

- Theorem:  $P(n) \equiv \sum_{0 \leq i \leq n} 2^i = 2^{n+1} - 1$
- Proof by induction
  - Basis:  $P(0) \equiv \sum_{0 \leq i \leq 0} 2^i = 2^{0+1} - 1 \equiv 2^0 = 2 - 1$ , which clearly holds
  - Step: We assume  $P(k) \equiv \sum_{0 \leq i \leq k} 2^i = 2^{k+1} - 1$  and show  $P(k+1) \equiv \sum_{0 \leq i \leq k+1} 2^i = 2^{k+2} - 1$  as follows:
 
$$\begin{aligned} \sum_{0 \leq i \leq k+1} 2^i &= \sum_{0 \leq i \leq k} 2^i + 2^{k+1} \\ &= (2^{k+1} - 1) + 2^{k+1} \text{ by the induction hypothesis} \\ &= 2 \cdot 2^{k+1} - 1 \\ &= 2^{k+2} - 1 \end{aligned}$$

24 October 2006

Winsborough CS 3233 Lecture 16

9

## Validity of Induction

- Induction is valid because the natural numbers are well founded
  - Definition: A set is *well founded* if each of its subsets has a least element
- Once basis and step are shown, the assumption that the property fails for some values yields a contradiction
  - Assume for contradiction that  $P(0) \wedge \forall k(P(k) \rightarrow P(k+1)) \wedge \neg \forall n P(n)$
  - Consider the least  $m \in \mathbb{N}$  such that  $\neg P(m)$ 
    - Case 1: if  $m = 0$ , the contradiction is immediate
    - Case 2: if  $m = k+1$  for some  $k \in \mathbb{N}$ , then by minimality of  $m$   $P(k)$  holds. But this, together with the step, implies that  $P(k+1) \equiv P(m)$  holds, giving us the desired contradiction

24 October 2006

Winsborough CS 3233 Lecture 16

10

## Strong Induction

- To show  $\forall n \in \mathbb{N} (P(n))$ , the following is sufficient:
  - Basis: show  $P(0)$
  - Inductive Step: show that for all  $k \in \mathbb{N}$ ,  $[P(0) \wedge \dots \wedge P(k)] \rightarrow P(k+1)$
- This gives us a stronger induction hypothesis to use in the step
- It is valid for similar reasons to those shown on the previous slide

24 October 2006

Winsborough CS 3233 Lecture 16

11