COT 5407: Introduction to Algorithms

Giri NARASIMHAN

Momentos

- Slides and Audio online
- Need to register
  
  - Go to [https://fiu.momentos.life](https://fiu.momentos.life)
  
  - If you don’t already have an account
    - Click on “Sign up”
    - Follow instructions & use referral code: XLY6FD
  
  - If you have an account, “Add Course” with course name and referral code XLY6FD
  
  - Verify account using link sent to email
Person of the Year ...
The first hundred votes ...

Who won a majority?

<table>
<thead>
<tr>
<th></th>
<th>48</th>
<th>93</th>
<th>93</th>
<th>93</th>
<th>12</th>
<th>12</th>
<th>93</th>
<th>12</th>
<th>93</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>48</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>23</td>
<td>12</td>
<td>22</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>93</td>
<td>48</td>
<td>48</td>
<td>12</td>
<td>93</td>
<td>12</td>
<td>33</td>
<td>79</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>23</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>33</td>
<td>93</td>
<td>93</td>
<td>93</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>23</td>
<td>93</td>
<td>48</td>
<td>48</td>
<td>12</td>
<td>12</td>
<td>44</td>
</tr>
<tr>
<td>93</td>
<td>93</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>93</td>
<td>12</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>48</td>
<td>12</td>
<td>48</td>
<td>48</td>
<td>12</td>
<td>48</td>
<td>88</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>93</td>
<td>12</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>33</td>
<td>93</td>
<td>93</td>
<td>93</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

Every number in the table corresponds to a vote for a person with that ID.

**Majority**: More than 50% of the votes
Standard Approaches

- Keep a list of candidates and their counts
  - Every vote needs to be compared against every candidate in the worst case
- Sort the list and count
  - Sorting is the bottleneck
  - Can we avoid sorting?
Wacky Ideas, anyone?

- What if I pick two random votes and they turn out to be different?
  - Discard and reduce the problem size
- What if I pick two random votes and they are the same?
  - Well, this needs work and you will need to think about it!
<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>48</td>
<td>12</td>
<td>9</td>
<td>12</td>
<td>23</td>
<td>12</td>
<td>22</td>
</tr>
<tr>
<td>48</td>
<td>93</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>93</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>93</td>
<td>48</td>
<td>48</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>33</td>
<td>93</td>
<td>93</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>9</td>
<td>12</td>
<td>23</td>
<td>93</td>
<td>48</td>
<td>12</td>
</tr>
<tr>
<td>93</td>
<td>93</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>48</td>
<td>12</td>
<td>48</td>
<td>48</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>93</td>
<td>12</td>
<td>12</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>33</td>
<td>93</td>
<td>93</td>
<td>12</td>
</tr>
</tbody>
</table>
Text

3rd Edition

- ISBN-10: 9788120340077
Evaluation

- Exams (2) 50%
- Quizzes 10%
- HW Assignments 30%
- Semester Project 5%
- Class Participation 5%
- Kattis Submissions 5% (Extra Credit)
Kattis

- Repository of problems
- Programming solutions can be uploaded
- Build a profile of problems solved by you
- Weekly mock competitions on Saturdays
  - E.g., FIU-SCIS-12JAN2019 (from noon to 5 PM)
What you should already know ...

- Array Lists
- Linked Lists
- Sorted Lists
- Stacks and Queues
- Basic Sorting Algorithms
- Trees
- Binary Search Trees
- Heaps and Priority Queues
- Graphs
  - Adjacency Lists
  - Adjacency Matrices
History of Algorithms

- Euclid, 300 BC
- Bhaskara, 6th c
- Al Khwarizmi, 9th c
- Fibonacci, 13th c
- Gauss, 18-19th c
- Babbage, 19th c
- Turing, 20th c
- von Neumann, 20th c
- Knuth, Karp, Tarjan, Rabin, …, 20-21st c
Gauss – sum of series

- \(1 + 2 + 3 + \ldots + N\)
- Gauss observed that
  - \(1 + N = N+1\)
  - \(2 + N-1 = N+1\)
  - \(\ldots\)
- Thus,
  - \(1 + 2 + 3 + \ldots + N\)
  - \(= (2 + 3 + \ldots + N-1) + (N+1)\)
  - \(= (3 + \ldots + N-2) + (N+1) + (N+1)\)
- Keep reducing until when?
  - Depends on whether \(N\) is even/odd
  - If \(N\) is even:
    - \(= (N+1) \frac{N}{2} = N(N+1)/2\)
  - If \(N\) is odd:
    - \(= (N+1) \frac{(N-1)}{2} + \frac{(N+1)}{2} = N(N+1)/2\)
Al Khwarizmi’s algorithm

43 x 17

43  17
21  34
10  68 (ignore)
  5  136
  2  272 (ignore)
  1  544

-------------------
    731
Euclid’s Algorithm

- GCD(12,8) = 4; GCD(49,35) = 7;
- GCD(210,588) = ??
- GCD(a,b) = ??
- **Observation**: [a and b are integers and \( a \geq b \)]
  - GCD(a,b) = GCD(a-b,b)
- **Euclid’s Rule**: [a and b are integers and \( a \geq b \)]
  - GCD(a,b) = GCD(a mod b, b)
- **Euclid’s GCD Algorithm**:
  - GCD(a,b)
    If \( b = 0 \) then return \( a \);
    return GCD(a mod b, b)
If you like Algorithms, nothing to worry about!

"Calculus is my new Versace. I get a buzz from algorithms. What's going on with me, Raymond? I'm scared."
You are asked to guess a number \( X \) that is known to be an integer lying in the range \( A \) through \( B \). How many guesses do you need in the worst case?

- Use binary search; Number of guesses = \( \log_2(B-A) \)

You are asked to guess a positive integer \( X \). How many guesses do you need in the worst case?

- NOTE: No upper bound is known for the number.
- Algorithm:
  - figure out \( B \) (by using Doubling Search)
  - perform binary search in the range \( B/2 \) through \( B \).
- Number of guesses = \( \log_2B + \log_2(B - B/2) \)
- Since \( X \) is between \( B/2 \) and \( B \), we have: \( \log_2(B/2) < \log_2X \),
- Number of guesses < \( 2\log_2X - 1 \)
Polynomial Evaluation

- Given a polynomial
  - \( p(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1} + a_n x^n \)
  - Compute the value of the polynomial for a given value of \( x \).
- How many additions and multiplications are needed?
  - **Simple solution:**
    - Number of additions = \( n \)
    - Number of multiplications = \( 1 + 2 + \ldots + n = n(n+1)/2 \)
  - **Reusing previous computations:** \( n \) additions and \( 2n \) multiplications!
  - **Improved solution using Horner’s rule:**
    - \( p(x) = a_0 + x(a_1 + x(a_2 + \ldots x(a_{n-1} + x a_n)))\ldots)) \)
    - Number of additions = \( n \)
    - Number of multiplications = \( n \)
Definitions

Abstract Problem: defines a function from any allowable input to a corresponding output

Instance of a Problem: a specific input to abstract problem

Algorithm: well-defined computational procedure that takes an instance of a problem as input and produces the correct output

An Algorithm must halt on every input with correct output.
Sorting

- Input is a sequence of \( n \) items that can be compared.
- Output is an ordered list of those \( n \) items
  - i.e., a reordering or permutation of the input items such that the items are in sorted order
- Fundamental problem that has received a lot of attention over the years.
- Used in many applications.
- Scores of different algorithms exist.
- Task: To compare algorithms
  - On what bases?
    - Time
    - Space
    - Other
Sorting Algorithms

- Number of Comparisons
- Number of Data Movements
- Additional Space Requirements
Sorting Algorithms

- SelectionSort
- InsertionSort
- BubbleSort
- ShakerSort
- MergeSort
- HeapSort
- QuickSort
- Bucket & Radix Sort
- Counting Sort