Happy New Year 2006!

Introduction and Basic Concepts

CS240: Data Structures and Data Management – Lecture 02

Jérémy Barbay

January 3, 2006

Happy New Year 2006!

Outline

When and Where

CS240
Tuesday, Thursday
10am-11:20am
MC 2054

Who

- Instructor: Jérémy Barbay

  email – jbarbay (at) uwaterloo.ca
  contact – DC 2332, x7824
  hours – Thursday 1-3pm

- Tutor: Olga Miltchman

  email – omiltchm (at) student.cs.uwaterloo.ca
  contact – DC ???
  hours – ???

- Administrative: Fenglian Qiu

  email – f2qiu (at) cs.uwaterloo.ca
  contact – DC 3115, Ext.2753

Electronic Resources

Web Page — `http://www.student.cs.uwaterloo.ca/~cs240`

- Before the class: slide handout.
- After the class: lecture summary.
- Every two weeks: a new assignment.
- Useful links and policies.

News Group — `news:uw.cs.cs240`

References

Introduction to Algorithms [CLRS]

- Covers 40% of our course material.
- Required readings for the course are given in the web page under the “Schedule” link.

Algorithm Design [Goodrich/Tamassia]

- Additional coverage for some specific topics.

Mark Breakdown

Prospective Mark Breakdown:

- Assignments 30%
- Midterm 30%
- Final Exam 40%

You pass the course iff your total average is ¿50%.

Assignments

- One assignment every two weeks, five in total.
- Hand-in and hand-out electronically.
- Release and retrieval on Tuesdays.
  Each assignment is worth 7 marks,
  1 mark removed per day late.
- Programming can be done in either Java or C++

All programming assignments will be tested in the Undergrad Math/CS Unix Environment.

Policies – Academic Discipline

University Policy 71 ("Student Academic Discipline Policy") contains relevant information and is available from the Web site of the University Secretariat at
`http://www.adm.uwaterloo.ca/infosec/`

- First offense:
  - -100% on the assignment,
  - at least -5% on your final course grade.
- Second offense: suspension for a term.

Summary

- All the information is on the webpage of the course at

  `http://www.student.cs.uwaterloo.ca/~cs240/`

- Regurly check the newsgroup to be informed of last minute changes.

  `news://uw.cs.cs240`

- See me or the tutor if necessary:
  - Jérémy Barbay, DC2332

Outline

Course Objectives.

- Sequel to CS134.
- Now focus on Data Structures, and Abstract Data Types.
- On the way:
  - Some more (light) mathematic analysis.
  - Some notions of the limits of computation.
  - some SQL.

Course Topics

- Our Computational Model
- Time and Space Analysis
- Lists
- Graphs
- Search Trees
- Priority Queues
- Hashing
- Text Compression
- Pattern Matching
- Sorting
- Database Systems
- Memory Management

Story Line

1. Analysis (Measuring Tape)
    - asymptotic Worst Case.
    - some Average Case.
2. Abstract Data Types (Kind of Tool: a hammer or a saw?)
    - Stack.
    - Queue.
    - Graph.
    - Tree.
    - Dictionary.
3. Data Structures (Material: Metal or Rubber hammer?)
    - array, matrix.
    - pointers, list.
    - hash
4. Applications (What to build: a chair or a table?)
    - Sorting
    - Text Compression
    - Text Search
    - Database
    - Memory Management

Summary

- Abstract Data Types ADT and Data Structures.
- From the Theory to the Applications.
- Some lower bounds.
- Some SQL.

Outline

Log and Exponent Identities

The more common identities you will likely use:

- $\log_b b^a = a$
- $b^{\log_b a} = a$
- $(b^a)^c = b^{ac}$
- $b^a b^c = b^{a+c}$
- $\log_b(ac) = \log_b a + \log_b c$
- $\log_b(a^c) = c \log_b a$
- $\log_b a = \frac{\log_c a}{\log_c b}$
- $b^{\log_c a} = a^{\log_c b}$

For short, note $\log_2 a$ as $\lg a$

Log/Exponent Identities (Cont')

- **Example**. Simplify:

$$\begin{aligned} \lg(2^n) + n^2 2^{3 \lg n} &= n + n^2 2^{3 \lg n} \\ &= n + n^2 n^3 \\ &= n + n^5 \\ &= n^5 + n \end{aligned}$$

- May need floors and ceilings:
  - $\lfloor 3.14159265 \rfloor = 3$
  - $\lceil 3.14159265 \rceil = 4$

Common Summations

- Arithmetic series:

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

Common Summations (cont')

- Useful approximation:

$$\sum_{i=1}^{n} i^k \approx \frac{n^{k+1}}{k+1}$$

- Geometric series (where $a \neq 1$):

$$\sum_{i=0}^{n} a^i = \frac{a^{n+1}-1}{a-1}$$

- Infinite series (where $0 < a < 1$):

$$\sum_{i=0}^{\infty} a^i = \frac{1}{1-a}$$

Derivations

Example:

$$\begin{aligned}\sum_{i=0}^{\infty} \frac{1}{2^i} &= \sum_{i=0}^{\infty} \left(\frac{1}{2}\right)^i \\ &= \frac{1}{1-\frac{1}{2}} \\ &= 2\end{aligned}$$

Factorial

- The number of arrangements of n distinct objects is $n!$.
- $n! = n \times (n-1) \times \cdots \times 2$
- Stirling's approximations:
  - $n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$
  - $\log(n!) \approx n \lg(n) - n + \frac{\ln n}{2} + \frac{\ln 2\pi}{2}$

Comparison of Algorithms

How do we find out which of these algorithms is the best?

- Selection Sort
- Merge Sort
- Counting Sort

Selection sort

```
for i:=1 to n
| min:=i;
| for j:=i+1 to n
| | if a[min]>a[j] min:=j;
| tmp:=a[i]; a[i]:=a[min]; a[min]:=tmp;
```

Merge sort

```
function sort(from,to)
| if (from>to)
| | mid:=floor(from+to/2);
| | sort(from,mid); sort(mid+1,to);
| | merge(from,mid,to);

function merge(from,mid,to)
| copy a[from..mid] to a new array b
| copy a[mid+1..to] to a new array c
| add infinity to both b and c as the last element
| k:=1; m:=1;
| for j:=from to to
| | if b[k]<c[m] then
| | | a[j]:=b[k]; k++;
| | else
| | | a[j]:=c[m]; m++;
```

Counting sort

Assume that all numbers in the array are between 1 and 1000:

```
clear array count[1..1000];
for i:=1 to n
| count[a[i]]++;
k:=0;
for i:=1 to 1000
| for j:=1 to count[i];
| | a[k]:=i; k++;
```

Comparison of Algorithms

How do we find out which of these algorithms is the best?

- Study worst/average case on instances of same size
- Measure time by key operations
- Suppose Uniform Time Access (RAM model)

Summary

- There are some Math Formula worth remembering.
- There are some algorithms worth studying.