Review

Chapter 4

Bahar Aameri
Department of Computer Science
University of Toronto

Mar 31, 2015

Announcements

- **Additional Instructor Office Hours (Bahar)**:
  - **Apr 01**, 12-1pm, BA3201
  - **Apr 08**, Time and location TBA.
- **Additional TA Office Hours**:
  - **Mar 30**, 4-6pm, BA3201
  - **Mar 31**, 4-6pm, BA3201
  - **Apr 01**, 1-3pm, BA2230
  - More to be announced!

The Final Exam

- **Content and Duration**:
  - Chapters 1.5, 2, 3, 4 (excluding Sections 4.2 and 4.3)
  - 3 Hours
- **Aid Sheet**:
  - **No** Aids are allowed
  - BUT, we will provide an **Aid Sheet**
- **The Aid Sheet will include:**
  - Derivation rules, **excluding** the following rules: Contrapositive, Implication, Equivalence, Double Negation, DeMorgan's, Implication Negation, Equivalence Negation, Quantifier Negation.
  - Definitions of Big-$\mathcal{O}$, Big-Ω, Big-Θ
  - Definitions of limit in the regular case and in the case when the limit is infinity.

The Final Exam

- **General Advise**:
  - Review the course notes.
  - Review the lectures notes.
  - Review all tutorial exercises and quizzes.
  - Review all your assignments and tests, identify where you made a mistake and find out why, review the sample solutions.
  - Do the exercises posted on the course website, and past exams.

Today's Topics

- **Disproving Bounds for Functions**
- **Algorithm Analysis**
- **Induction**

Disproving Upper Bound using Limits

$\frac{f(n)}{g(n)} > c$

Reminder: Big-$\mathcal{O}$

- $\mathbf{f} \in \mathcal{O}(\mathbf{g})$:
  $\exists \mathbf{c} \in \mathbb{R}^+, \exists \mathbf{B} \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq \mathbf{B} \Rightarrow f(n) \leq \mathbf{c}.g(n)$
- $\mathbf{f} \notin \mathcal{O}(\mathbf{g})$:
  $\forall \mathbf{c} \in \mathbb{R}^+, \forall \mathbf{B} \in \mathbb{N}, \exists n \in \mathbb{N}, n \geq \mathbf{B} \wedge f(n) > \mathbf{c}.g(n)$

Reminder: Special Case of Limits

- $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$
  $\iff$
  $\forall \varepsilon \in \mathbb{R}^+, \exists n' \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq n' \implies \frac{f(n)}{g(n)} > \varepsilon$

Disproving Upper Bound using Limits

Reminder: Special Case of Limits

- $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$
  $\iff$
  $\forall c \in \mathbb{R}^+, \exists n' \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq n' \implies f(n) > c.g(n)$

- $\forall c \in \mathbb{R}^+, \forall B \in \mathbb{N}, \exists n \in \mathbb{N}, n \geq B \wedge f(n) > c.g(n)$

Assume $c \in \mathbb{R}^+$, assume $B \in \mathbb{N}$. # arbitrary values

Then $\exists n' \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq n' \implies f(n) > c.g(n)$. # definition of $\lim_{n\to\infty} f(n)/g(n) = \infty$

Let n_1 be such that $\forall n \in \mathbb{N}, n \geq n_1 \implies f(n) > c.g(n)$. # instantiate n'

Let $n_0 = \max(B, n_1)$. Then $n_0 \in \mathbb{N}$.

Then $n_0 \geq B$. # by definition of max

Then $f(n_0) > c.g(n_0)$. # by the assumption above $f(n) > c.g(n)$, since $n_0 \geq n_1$

Then $n_0 \geq B \wedge f(n_0) \geq c.g(n_0)$. # introduce $\wedge$

Then $\exists n \in \mathbb{N}, n \geq B \wedge f(n) > c.g(n)$ # introduce $\exists$

Then $\forall c \in \mathbb{R}, \forall B \in \mathbb{N}, \exists n \in \mathbb{N}, n \geq B \wedge f(n) > c.g(n)$ # introduce $\forall$

Disproving Upper Bound using Limits

Disproving Big-$\mathcal{O}$

- Show that $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$.
- Use the proof in the previous slide to show that $\mathbf{f} \notin \mathcal{O}(\mathbf{g})$.

Disproving Lower Bounds

$$\frac{f(n)}{g(n)} \geq c$$

Reminder: Big-Ω

- $\mathbf{f} \in \mathbf{\Omega}(\mathbf{g})$:
  $\exists \mathbf{c} \in \mathbb{R}^+, \exists \mathbf{B} \in \mathbb{N}, \forall n \in \mathbb{N}, n \geq \mathbf{B} \Rightarrow f(n) \geq \mathbf{c}.g(n)$
- $\mathbf{f} \notin \mathbf{\Omega}(\mathbf{g})$:
  $\forall \mathbf{c} \in \mathbb{R}^+, \forall \mathbf{B} \in \mathbb{N}, \exists n \in \mathbb{N}, n \geq \mathbf{B} \wedge f(n) < \mathbf{c}.g(n)$

Reminder: Limits

- $\lim_{n\to\infty} \frac{f(n)}{g(n)} = L$
  $\iff$
  $\forall \varepsilon \in \mathbb{R}^+, \exists \mathbf{n}' \in \mathbb{N}, \forall \mathbf{n} \in \mathbb{N}, \mathbf{n} \geq \mathbf{n}' \Rightarrow \mathbf{L} - \varepsilon < \frac{\mathbf{f(n)}}{\mathbf{g(n)}} < \mathbf{L} + \varepsilon$
- $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$
  $\iff$
  $\forall \varepsilon \in \mathbb{R}^+, \exists \mathbf{n}' \in \mathbb{N}, \forall \mathbf{n} \in \mathbb{N}, \mathbf{n} \geq \mathbf{n}' \Rightarrow -\varepsilon < \frac{\mathbf{f(n)}}{\mathbf{g(n)}} < \varepsilon$

Proving a Tight Bound $W(n)$ for Worst-case Running Time

- Give an expression $U(n)$ which represents an **overestimate** of the worst-case running time.
- Give an specific case such that the corresponding running time function $L(n)$ is in $\Omega(W)$.
- Prove $L \in \Omega(W)$ and $U \in \mathcal{O}(W)$

Algorithm Analysis

1+n+n+4n+3n+3n+n=13n+1

```
def mystery2(L):
""" L is a non-empty list of length len(L) = n. """
    i = 1        1                                   # line  1

    while i < len(L) - 1:    n                       # line  2

        j = i - 1            n                       # line  3

        while j <= i + 1:     4n                     # line  4  i+1-(i-1)+1+1=4

            L[j] = L[j] + L[i]    3n                 # line  5

            j = j + 1             3n                 # line  6

        i = i + 1            n                       # line  7
```

Algorithm Analysis

$$2+n+1+n+n^2+n+2n^2+n^3+n^2+2n^3+3n^2+n+1=\mathbf{4+4n+7n^2+3n^3}$$

Running Time Analysis for Maximum Sum (MS)

- **Claims**: $T_{MS} \in \mathcal{O}(n^3)$

```
def max_sum(L):
    # To generate all non-empty slices [i:j] for list L, i must
    # take on values from 0 to len(L)-1, and j must take on
    # values from i+1 to len(L).
    max = 0      1                                  # line  1
    i = 0        1                                  # line  2
    while i < len(L):   n+1                         # line  3
        j = i + 1       n                           # line  4
        while j <= len(L): n(n-i+1) =< n^2+n        # line  5
            # Compute the sum of L[i:j].
            sum = 0     n^2                         # line  6
            k = i       n^2                         # line  7
            while k < j: n^2(j-i+1) =< n^3+n^2      # line  8
                sum = sum + L[k]   n^3              # line  9
                k = k + 1          n^3              # line 10
            # Update max if appropriate.
            if sum > max:   n^2                     # line 11
                max = sum   n^2                     # line 12
            j = j + 1       n^2                     # line 13
        i = i + 1   n                               # line 14
    # At this point, we've examined every slice.
    return max   1                                  # line 15
```

We only consider the first 1/3 iteration of the loop over i

Running Time Analysis for Maximum Sum (MS)

- **Claims**: $T_{MS} \in \Omega(n^3)$

```
def max_sum(L):
    # To generate all non-empty slices [i:j] for list L, i must
    # take on values from 0 to len(L)-1, and j must take on
    # values from i+1 to len(L).
    max = 0                                          # line  1
    i = 0                                            # line  2
    while i < len(L): n/3                            # line  3
        j = i + 1                                    # line  4
        while j <= len(L):n/3(n-n/3) = 2n^2/9        # line  5
            # Compute the sum of L[i:j].
            sum = 0                                  # line  6
            k = i                                    # line  7
            while k < j:n^2/9(2n/3-n/3)=n^3/27       # line  8
                sum = sum + L[k]                     # line  9
                k = k + 1                            # line 10
            # Update max if appropriate.
            if sum > max:                            # line 11
                max = sum                            # line 12
            j = j + 1                                # line 13
        i = i + 1                                    # line 14
    # At this point, we've examined every slice.
    return max                                       # line 15
```

at least $n^3/27$ steps

Proof by Induction

Prove $\forall n \in \mathbb{N} \setminus S, P(n)$

- Prove the **Base Case**, P(b).
- Assume P(n) (**Induction Hypothesis**), and prove $P(n+1)$.

Proof by Induction

An Easy Exercise

- Show that $1 + 2 + ... + n = \frac{n(n+1)}{2}$