Introduction to CS 4104

T. M. Murali

August 24, 2009
Course Information

▶ Instructor
  ▶ T. M. Murali, 2160B Torgerson, 231-8534, murali@cs.vt.edu
  ▶ Office Hours: 9:30am–11:30am Mondays and Wednesdays

▶ Teaching assistant
  ▶ Chris Poirel, poirel@vt.edu
  ▶ Office Hours: to be decided
Course Information

▶ Instructor
  - T. M. Murali, 2160B Torgerson, 231-8534, murali@cs.vt.edu
  - Office Hours: 9:30am–11:30am Mondays and Wednesdays

▶ Teaching assistant
  - Chris Poirel, poirel@vt.edu
  - Office Hours: to be decided

▶ Class meeting time
  - MW 2:30pm–3:45pm, McBryde 134
Course Information

- Instructor
  - T. M. Murali, 2160B Torgerson, 231-8534, murali@cs.vt.edu
  - Office Hours: 9:30am–11:30am Mondays and Wednesdays

- Teaching assistant
  - Chris Poirel, poirel@vt.edu
  - Office Hours: to be decided

- Class meeting time
  - MW 2:30pm–3:45pm, McBryde 134

- Keeping in Touch
  - Course web site
    - http://courses.cs.vt.edu/~cs4104/spring2009, updated regularly through the semester
  - Listserv: cs4104_91844@listserv.vt.edu
Course Information

▶ Instructor
  ▶ T. M. Murali, 2160B Torgerson, 231-8534, murali@cs.vt.edu
  ▶ Office Hours: 9:30am–11:30am Mondays and Wednesdays

▶ Teaching assistant
  ▶ Chris Poiriel, poirel@vt.edu
  ▶ Office Hours: to be decided

▶ Class meeting time
  ▶ MW 2:30pm–3:45pm, McBryde 134

▶ Keeping in Touch
  ▶ Course web site
    http://courses.cs.vt.edu/~cs4104/spring2009, updated regularly through the semester
  ▶ Listserv: cs4104_91844@listserv.vt.edu

▶ Prerequisites: CS 2604 or CS 2606, MATH 3134 or MATH 3034
Required Course Textbook

- Algorithm Design
- Jon Kleinberg and Éva Tardos
- Addison-Wesley
- 2006
Course Goals

▶ Learn methods and principles to construct algorithms.
▶ Learn techniques to analyze algorithms mathematically for correctness and efficiency (e.g., running time and space used).
▶ Course roughly follows the topics suggested in textbook
  ▶ Measures of algorithm complexity
  ▶ Graphs
  ▶ Greedy algorithms
  ▶ Divide and conquer
  ▶ Dynamic programming
  ▶ Network flow problems
  ▶ NP-completeness
Required Readings

- Reading assignment available on the website.
- Read **before** class.
Lecture Slides

- Will be available on class web site.
- Usually posted just before class.
- Class attendance is extremely important.
Lecture Slides

- Will be available on class web site.
- Usually posted just before class.
- **Class attendance is extremely important.** Lecture in class contains significant and substantial additions to material on the slides.
Homeworks

- Posted on the web site ≈ one week before due date.
- Announced on the class listserv.
- Prepare solutions digitally but hand in hard-copy.
Homeworks

- Posted on the web site ≈ one week before due date.
- Announced on the class listserv.
- Prepare solutions digitally but hand in hard-copy.
  - Solution preparation recommended in $\LaTeX$. 
Examinations

- Take-home midterm.
- Take-home final (comprehensive).
- Prepare digital solutions (recommend \LaTeX).
Examinations

- Take-home midterm.
- Take-home final (comprehensive).
- Prepare digital solutions (recommend \LaTeX).
- Examinations may change to be in class.
Grades

- Homeworks: \approx 8, 60\% of the grade.
- Take-home midterm: 15\% of the grade.
- Take-home final: 25\% of the grade.
What is an Algorithm?
What is an Algorithm?

Chamber’s  A set of prescribed computational procedures for solving a problem; a step-by-step method for solving a problem.

Knuth, TAOCP  An algorithm is a finite, definite, effective procedure, with some input and some output.
Origin of the word “Algorithm”

1. From the Arabic *al-Khwarizmi*, a native of Khwarazm, a name for the 9th century mathematician, Abu Ja’far Mohammed ben Musa.
# Origin of the word “Algorithm”

1. From the Arabic *al-Khwarizmi*, a native of Khwarazm, a name for the 9th century mathematician, Abu Ja’far Mohammed ben Musa.

2. From Al Gore, the former U.S. vice-president who invented the internet.
Origin of the word “Algorithm”

1. From the Arabic *al-Khwārizmi*, a native of Khwarazm, a name for the 9th century mathematician, Abu Ja’far Mohammed ben Musa.

2. From Al Gore, the former U.S. vice-president who invented the internet.

3. From the Greek *algos* (meaning “pain,” also a root of “analgesic”) and *rythmos* (meaning “flow,” also a root of “rhythm”).
Origin of the word “Algorithm”

1. From the Arabic *al-Khwarizmi*, a native of Khwarazm, a name for the 9th century mathematician, Abu Ja’far Mohammed ben Musa.

2. From Al Gore, the former U.S. vice-president who invented the internet.

3. From the Greek *algos* (meaning “pain,” also a root of “analgesic”) and *rythmos* (meaning “flow,” also a root of “rhythm”). “Pain flowed through my body whenever I worked on CS 4104 homeworks.” – former student.
Origin of the word “Algorithm”

1. From the Arabic *al-Khwarizmi*, a native of Khwarazm, a name for the 9th century mathematician, Abu Ja’far Mohammed ben Musa. He wrote “Kitab al-jabr wa’l-muqabala,” which evolved into today’s high school algebra text.

2. From Al Gore, the former U.S. vice-president who invented the internet.

3. From the Greek *algos* (meaning “pain,” also a root of “analgesic”) and *rythmos* (meaning “flow,” also a root of “rhythm”). “Pain flowed through my body whenever I worked on CS 4104 homeworks.” – former student.
Problem Example

Find Minimum

INSTANCE: Nonempty list $x_1, x_2, \ldots, x_n$ of integers.

SOLUTION: Pair $(i, x_i)$ such that $x_i = \min\{x_j \mid 1 \leq j \leq n\}$. 

Algorithm Example

Find-Minimum($x_1, x_2, \ldots, x_n$)
1 \hspace{1em} i \leftarrow 1
2 \hspace{1em} \textbf{for} j \leftarrow 2 \ \textbf{to} \ n
3 \hspace{1em} \hspace{1em} \textbf{do} \hspace{1em} \textbf{if} \ x_j < x_i
4 \hspace{1em} \hspace{1em} \hspace{1em} \textbf{then} \ i \leftarrow j
5 \hspace{1em} \textbf{return} \ (i, x_i)
Running Time of Algorithm

Find-Minimum\((x_1, x_2, \ldots, x_n)\)

1 \hspace{1em} i \leftarrow 1

2 \hspace{1em} \text{for } j \leftarrow 2 \text{ to } n

3 \hspace{1em} \text{do if } x_j < x_i

4 \hspace{1em} \text{then } i \leftarrow j

5 \hspace{1em} \text{return } (i, x_i)
Running Time of Algorithm

Find-Minimum($x_1, x_2, \ldots, x_n$)
1 $i \leftarrow 1$
2 for $j \leftarrow 2$ to $n$
3     do if $x_j < x_i$
4         then $i \leftarrow j$
5 return $(i, x_i)$

- At most $2n - 1$ assignments and $n - 1$ comparisons.
Correctness of Algorithm: Proof 1

Find-Minimum\((x_1, x_2, \ldots, x_n)\)

1. \(i \leftarrow 1\)
2. for \(j \leftarrow 2\) to \(n\)
3. do if \(x_j < x_i\)
4. \(\text{then } i \leftarrow j\)
5. return \((i, x_i)\)
Correctness of Algorithm: Proof 1

Find-Minimum\((x_1, x_2, \ldots, x_n)\)

1 \hspace{1cm} i \leftarrow 1
2 \hspace{1cm} \textbf{for} \ j \leftarrow 2 \ \textbf{to} \ n \\
3 \hspace{1cm} \hspace{1cm} \textbf{do} \ \textbf{if} \ x_j < x_i \\
4 \hspace{1cm} \hspace{1cm} \hspace{1cm} \textbf{then} \ i \leftarrow j \\
5 \hspace{1cm} \textbf{return} \ (i, x_i)

▸ Proof by contradiction:
Correctness of Algorithm: Proof 1

Find-Minimum($x_1, x_2, \ldots, x_n$)
1 $i \leftarrow 1$
2 for $j \leftarrow 2$ to $n$
3     do if $x_j < x_i$
4         then $i \leftarrow j$
5 return $(i, x_i)$

Proof by contradiction: Suppose algorithm returns $(k, x_k)$ but there exists $1 \leq l \leq n$ such that $x_l < x_k$ and $x_l = \min\{x_j \mid 1 \leq j \leq n\}$.
Correctness of Algorithm: Proof 1

Find-Minimum($x_1, x_2, \ldots, x_n$)
1  \( i \leftarrow 1 \)
2  \( \text{for } j \leftarrow 2 \text{ to } n \)
3  \( \quad \text{do if } x_j < x_i \)
4  \( \quad \text{then } i \leftarrow j \)
5  \( \text{return } (i, x_i) \)

- Proof by contradiction: Suppose algorithm returns \((k, x_k)\) but there exists \(1 \leq l \leq n\) such that \(x_l < x_k\) and \(x_l = \min\{x_j \mid 1 \leq j \leq n\}\).
- Is \(k < l\)?
Correctness of Algorithm: Proof 1

Find-Minimum($x_1, x_2, \ldots, x_n$)
1 $i \leftarrow 1$
2 $\text{for } j \leftarrow 2 \text{ to } n$
3 \hspace{1em} $\text{do if } x_j < x_i$
4 \hspace{2em} $\text{then } i \leftarrow j$
5 $\text{return } (i, x_i)$

- Proof by contradiction: Suppose algorithm returns $(k, x_k)$ but there exists $1 \leq l \leq n$ such that $x_l < x_k$ and $x_l = \min\{x_j \mid 1 \leq j \leq n\}$.
- Is $k < l$? No. Since the algorithm returns $(k, x_k)$, $x_k \leq x_j$, for all $k < j \leq n$. Therefore $l < k$. 
Correctness of Algorithm: Proof 1

Find-Minimum($x_1, x_2, \ldots, x_n$)

1. $i \leftarrow 1$
2. for $j \leftarrow 2$ to $n$
   3. do if $x_j < x_i$
      4. then $i \leftarrow j$
3. return $(i, x_i)$

- Proof by contradiction: Suppose algorithm returns $(k, x_k)$ but there exists $1 \leq l \leq n$ such that $x_l < x_k$ and $x_l = \min \{x_j \mid 1 \leq j \leq n\}$.
- Is $k < l$? No. Since the algorithm returns $(k, x_k)$, $x_k \leq x_j$, for all $k < j \leq n$. Therefore $l < k$.
- What does the algorithm do when $j = l$? It must set $i$ to $l$, since we have been told that $x_l$ is the smallest element.
- What does the algorithm do when $j = k$ (which happens after $j = l$)? Since $x_l < x_k$, the value of $i$ does not change.
- Therefore, the algorithm does not return $(k, x_k)$ yielding a contradiction.
Correctness of Algorithm: Proof 2

Find-Minimum($x_1, x_2, \ldots, x_n$)
1    $i \leftarrow 1$
2    for $j \leftarrow 2$ to $n$
3       do if $x_j < x_i$
4          then $i \leftarrow j$
5    return $(i, x_i)$

Proof by induction: What is true at the end of each iteration?
Correctness of Algorithm: Proof 2

Find-Minimum\((x_1, x_2, \ldots, x_n)\)

1. \(i \leftarrow 1\)
2. \(\text{for } j \leftarrow 2 \text{ to } n\)
3. \(\quad \text{do if } x_j < x_i\)
4. \(\quad \text{then } i \leftarrow j\)
5. \(\text{return } (i, x_i)\)

▶ Proof by induction: What is true at the end of each iteration?
▶ Claim: \(x_i = \min\{x_m \mid 1 \leq m \leq j\}\), for all \(1 \leq j \leq n\).
▶ Claim is true
Correctness of Algorithm: Proof 2

Find-Minimum($x_1, x_2, \ldots, x_n$)

1. $i \leftarrow 1$
2. for $j \leftarrow 2$ to $n$
3.  do if $x_j < x_i$
4.  then $i \leftarrow j$
5. return $(i, x_i)$

Proof by induction: What is true at the end of each iteration?

Claim: $x_i = \min\{x_m \mid 1 \leq m \leq j\}$, for all $1 \leq j \leq n$.

Claim is true $\Rightarrow$ algorithm is correct (set $j = n$).
Correctness of Algorithm: Proof 2

Find-Minimum($x_1, x_2, \ldots, x_n$)

1. $i \leftarrow 1$
2. for $j \leftarrow 2$ to $n$
   3. do if $x_j < x_i$
      4. then $i \leftarrow j$
3. return $(i, x_i)$

 Proof by induction: What is true at the end of each iteration?

Claim: $x_i = \min \{x_m \mid 1 \leq m \leq j\}$, for all $1 \leq j \leq n$.

Claim is true $\Rightarrow$ algorithm is correct (set $j = n$).

Proof of the claim involves three steps.

1. Base case: $j = 1$ (before loop). $x_i = \min \{x_m \mid 1 \leq m \leq 1\}$ is trivially true.
2. Inductive hypothesis: Assume $x_i = \min \{x_m \mid 1 \leq m \leq j\}$.
3. Inductive step: Prove $x_i = \min \{x_m \mid 1 \leq m \leq j + 1\}$.
   - In the loop, $i$ is set to be $j + 1$ if and only if $x_{j+1} < x_i$.
   - Therefore, $x_i$ is the smallest of $x_1, x_2, \ldots, x_{j+1}$ after the loop.
Format of Proof by Induction

- Goal: prove some proposition $P(n)$ is true for all $n$.
- Strategy: prove base case, assume inductive hypothesis, prove inductive step.
Format of Proof by Induction

▶ Goal: prove some proposition $P(n)$ is true for all $n$.
▶ Strategy: prove base case, assume inductive hypothesis, prove inductive step.
▶ Base case: prove that $P(1)$ or $P(2)$ (or $P$(small number)) is true.
▶ Inductive hypothesis: assume $P(n - 1)$ is true.
▶ Inductive step: prove that $P(n - 1) \Rightarrow P(n)$.
Format of Proof by Induction

- **Goal:** prove some proposition $P(n)$ is true for all $n$.
- **Strategy:** prove base case, assume inductive hypothesis, prove inductive step.
- **Base case:** prove that $P(1)$ or $P(2)$ (or $P$(small number)) is true.
- **Inductive hypothesis:** assume $P(n-1)$ is true.
- **Inductive step:** prove that $P(n-1) \Rightarrow P(n)$.
- **Why does this strategy work?**
Sum of first $n$ natural numbers

$$P(n) = \sum_{i=1}^{n} i =$$
Sum of first $n$ natural numbers

$$P(n) = \sum_{i=1}^{n} i = \frac{n(n+1)}{2}.$$
Recurrence Relation

Given

\[
P(n) = \begin{cases} 
P\left(\left\lfloor \frac{n}{2} \right\rfloor \right) + 1 & \text{if } n > 1 \\
1 & \text{if } n = 1
\end{cases}
\]

prove that

\[P(n) \leq \]
Recurrence Relation

Given

\[ P(n) = \begin{cases} 
  P(\lfloor \frac{n}{2} \rfloor) + 1 & \text{if } n > 1 \\
  1 & \text{if } n = 1 
\end{cases} \]

prove that

\[ P(n) \leq 1 + \log_2 n. \]
Recurrence Relation

Given

\[ P(n) = \begin{cases} 
P(\lfloor \frac{n}{2} \rfloor) + 1 & \text{if } n > 1 \\
1 & \text{if } n = 1 
\end{cases} \]

prove that

\[ P(n) \leq 1 + \log_2 n. \]

- Use strong induction: In the inductive hypothesis, assume that \( P(i) \) is true for all \( i < n \).