CSCM12: software concepts and efficiency
What's an algorithm?

Cécilia Pradic

Swansea University, 30/01/2024

Recommended reading after this lecture

Just the introduction to one of the textbooks in the reading list!

Definition

- Have you even heard the word algorithm?
- What do you think this means?

Definition

- Have you even heard the word algorithm?
- What do you think this means?

Some definition, ∼wiktionary

A *precise step-by-step* plan for a computational procedure that possibly begins with some inputs and yields outputs in a **finite number of steps**.

Definition

- Have you even heard the word algorithm?
- What do you think this means?

Some definition, ∼wiktionary

A *precise step-by-step* plan for a computational procedure that possibly begins with some inputs and yields outputs in a **finite number of steps**.

- tentative, lots of rooms to quibble
    - what does precise mean?

Definition

- Have you even heard the word algorithm?
- What do you think this means?

Some definition, ∼wiktionary

A *precise step-by-step* plan for a computational procedure that possibly begins with some inputs and yields outputs in a **finite number of steps**.

- tentative, lots of rooms to quibble
    - what does precise mean?
    - what steps?

Definition

- Have you even heard the word algorithm?
- What do you think this means?

Some definition, ∼wiktionary

A *precise step-by-step* plan for a computational procedure that possibly begins with some inputs and yields outputs in a **finite number of steps**.

- tentative, lots of rooms to quibble
    - what does precise mean?
    - what steps?
    - what language?

Definition

- Have you even heard the word algorithm?
- What do you think this means?

Some definition, ∼wiktionary

A *precise step-by-step* plan for a computational procedure that possibly begins with some inputs and yields outputs in a **finite number of steps**.

- tentative, lots of rooms to quibble
  - what does precise mean?
  - what steps?
  - what language?
  - what computational model?

Definition

- Have you even heard the word algorithm?
- What do you think this means?

Some definition, ∼wiktionary

A *precise step-by-step* plan for a computational procedure that possibly begins with some inputs and yields outputs in a **finite number of steps**.

- tentative, lots of rooms to quibble
  - what does precise mean?
  - what steps?
  - what language?
  - what computational model?
- etymology: after al-Kwarizmi

(Persian mathematician, 9th century AD)

“Example”: cooking recipe

Input: eggs, oranges, sugar, UK

Output: moudly cake

```
Begin by preheating the oven to 180C and greasing the pan.
In a separate medium-sized bowl, combine the flour, baking
Next, using a mixer fitted with the whisk attachment, beat
With the mixer running on low speed, slowly add the oil and
Add batter to the prepared pan. Bake 50 to 60 minutes for a
Allow to cool for 30 minutes in the pan before turning out
```

Example 2: some maths

- You have definitely learned some algorithms in maths!
    - Long addition (compute by hand $545256 + 687622$, division?
    - Simplifying fractions
    - Solving equations

Solving quadratic equations

- Input: real numbers a, b, c with $a \neq 0$?
- Output: nice expressions for the set of (real-valued) roots
- Algorithm:
    1. Compute $\Delta = b^2 - 4ac$
    2. If $\Delta < 0$, output $\emptyset$
    3. If $\Delta = 0$, output $\left\{-\frac{b}{2a}\right\}$
    4. Otherwise, output $\left\{\frac{-b+\sqrt{\Delta}}{2a}, \frac{-b-\sqrt{\Delta}}{2a}\right\}$

Running the algorithm step-by-step

Solving quadratic equations

- Input: real numbers a, b, c with $a \neq 0$?
- Output: nice expressions for the set of (real-valued) roots
- Algorithm:
  1. Compute $\Delta = b^2 - 4ac$
  2. If $\Delta < 0$, output $\emptyset$
  3. If $\Delta = 0$, output $\left\{-\frac{b}{2a}\right\}$
  4. Otherwise, output $\left\{\frac{-b+\sqrt{\Delta}}{2a}, \frac{-b-\sqrt{\Delta}}{2a}\right\}$

Running the algorithm step-by-step

Solving quadratic equations

- Input: real numbers a, b, c with $a \neq 0$?
- Output: nice expressions for the set of (real-valued) roots
- Algorithm:
  1. Compute $\Delta = b^2 - 4ac$
  2. If $\Delta < 0$, output $\emptyset$
  3. If $\Delta = 0$, output $\left\{-\frac{b}{2a}\right\}$
  4. Otherwise, output $\left\{\frac{-b+\sqrt{\Delta}}{2a}, \frac{-b-\sqrt{\Delta}}{2a}\right\}$

- For $a = b = c = 1$?

Running the algorithm step-by-step

Solving quadratic equations

- Input: real numbers a, b, c with $a \neq 0$?
- Output: nice expressions for the set of (real-valued) roots
- Algorithm:

```
1. Compute $\Delta = b^2 - 4ac$
2. If $\Delta < 0$, output $\emptyset$
3. If $\Delta = 0$, output $\left\{-\frac{b}{2a}\right\}$
4. Otherwise, output $\left\{\frac{-b+\sqrt{\Delta}}{2a}, \frac{-b-\sqrt{\Delta}}{2a}\right\}$
```

- For $a = b = c = 1$?
- For $a = c = 1$, $b = 2$?

Running the algorithm step-by-step

Solving quadratic equations

- Input: real numbers a, b, c with $a \neq 0$?
- Output: nice expressions for the set of (real-valued) roots
- Algorithm:

```
1. Compute $\Delta = b^2 - 4ac$
2. If $\Delta < 0$, output $\emptyset$
3. If $\Delta = 0$, output $\left\{-\frac{b}{2a}\right\}$
4. Otherwise, output $\left\{\frac{-b+\sqrt{\Delta}}{2a}, \frac{-b-\sqrt{\Delta}}{2a}\right\}$
```

- For $a = b = c = 1$?
- For $a = c = 1$, $b = 2$?
- For $a = 1, b = 5, c = 2$?

Running the algorithm step-by-step

Solving quadratic equations

- Input: real numbers a, b, c with $a \neq 0$?
- Output: nice expressions for the set of (real-valued) roots
- Algorithm:
  1. Compute $\Delta = b^2 - 4ac$
  2. If $\Delta < 0$, output $\emptyset$
  3. If $\Delta = 0$, output $\left\{-\frac{b}{2a}\right\}$
  4. Otherwise, output $\left\{\frac{-b+\sqrt{\Delta}}{2a}, \frac{-b-\sqrt{\Delta}}{2a}\right\}$

- For $a = b = c = 1$?
- For $a = c = 1$, $b = 2$?
- For $a = 1, b = 5, c = 2$?

Some assumptions you may question

- What is a **step**?
- What is the cost of a **step**?

Some assumptions you may question

- What is a **step**?
- What is the cost of a **step**?

Answer

Depends on the context!

- I will give you more details later, *for algorithms meant to be ran on computers*.

Dictionary search

- Goal: look up the definition of a word in a dicti
- Basic steps:
    - open the dictionary at a given page number
    - read all of the data on a page
    - (basic arithmetic, etc)

Dictionary search

- Goal: look up the definition of a word in a dicti
- Basic steps:
  - open the dictionary at a given page number
  - read all of the data on a page
  - (basic arithmetic, etc)

What do you think of the following method?

1. Begin at page $n = 0$ (the cover)
2. Check if the definition is at page n
   - 2.1 If yes, return the definition
   - 2.2 If no, go to the next page (turn n into $n + 1$) and go to step 2.

Dictionary search

- Goal: look up the definition of a word in a dicti
- Basic steps:
  - open the dictionary at a given page number
  - read all of the data on a page
  - (basic arithmetic, etc)

What do you think of the following method?

1. Begin at page $n = 0$ (the cover)
2. Check if the definition is at page n
   - 2.1 If yes, return the definition
   - 2.2 If no, go to the next page (turn n into $n + 1$) and go to step 2.

How many steps does it cost?

The one trick you already know

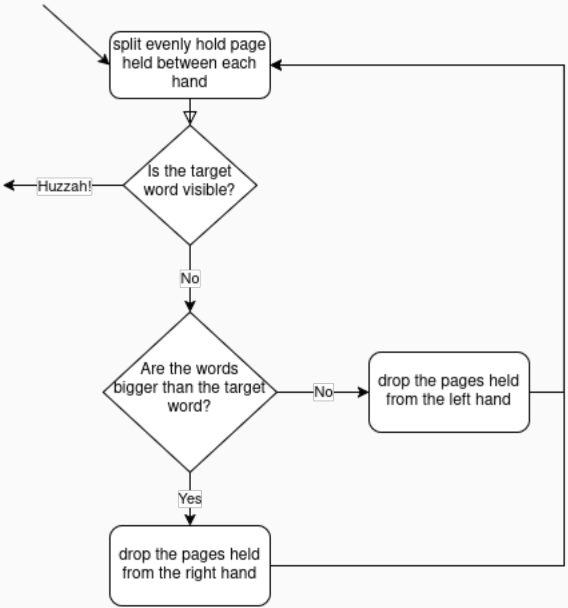
Dichotomy search

1. Look at the total number of pages, remember that as n
2. Set $k = 0$
3. Go to page $m = \lfloor k + \frac{n}{2} \rfloor$
    3.1 if it contains the word, good!
    3.2 if the word at the top of the page is *bigger in the lexicographic order*
        - set $n = m$, go back to step 3.
    3.3 otherwise
        - set $k = m$, go back to step 3.

Different ways of representing algorithms

- informal text (what we did so far)
- graphical (flowcharts, graphical programming languages)
- code, pseudo-code (= informal version of computer programs)

Flowcharts


split evenly hold page held between each hand
Huzzah!
Is the target word visible?
No
Are the words bigger than the target word?
No
drop the pages held from the left hand
Yes
drop the pages held from the right hand

Code

- Pro: rather rigorous, can run on computer
- Con: sometimes a bit for humans

```
/* Assumptions: arr contains an increasing
                sequence of values
                arr[mi] <= 0 and arr[ma] >=0*/
static int dicho_rec(int[] arr, int mi, int ma)
{
   if (ma <= mi)
      return mi;
   final int mid = (ma+mi)/2;
   if (arr[mid] <= 0)
     return dicho_rec(arr,mid,ma);
   else
     return dicho_rec(arr,mi,mid);
}
```

What is pseudo-code?

- More informal version of computer code
- Should be first and foremost human-readable

```
FindIndexDicho($A, x$)
    $start \leftarrow 0$
    $end \leftarrow$ size of $A$
    while $start < end$ do
        $mid \leftarrow \lceil \frac{end+start}{2} \rceil$
        if $A[mid] \leq x$ then
            $start \leftarrow mid$
        else
            $end \leftarrow mid$

    if $A[start] = x$ then
        return $start$
    else
        return $-1$
```

How will we run algorithms?

- By hand
- By implementing them on the computer

How will we run algorithms?

- By hand
- By implementing them on the computer

Big question of this module

How much resources is required to run an algorithm?

- Time?
- Memory?

In the next episode

In the lab

- You will write your own algorithm
- You will measure and compare the **running time** in practice of java functions implementing the same functionality

(baby benchmarks)

In the next lecture

I will show you how to estimate the complexity of algorithms

- rough **a priori** analyses

(i.e. without implementing and benchmarking)

- discard methods that much worse than others at a glance