Algorithms do change the world

John MacCormick
Dickinson College
Outline: life and learning in the age of algorithms

• Are we living in an age of algorithms?

• What is an algorithm?

• What algorithmic ideas should be in the school-age maths curriculum?
Are we living in an age of algorithms?

Mentions of "algorithm" in New York Times

visit Princeton for initial marketing discussions

“The word ‘algorithm’ is too scary; we can’t use it in the title”

The Social Network: “I need the algorithm you used to rank chess players”

New title idea from marketing department...
... “9 Algorithms That Changed the Future”!
What is an algorithm?

“at first glance it may look as though someone intended to write ‘logarithm’ but jumbled up the first four letters”

Donald E. Knuth (1968), *The Art of Computer Programming*
What is an algorithm?

• Please take 30 seconds to come up with your own definition
What is an algorithm?

Source: top 15 definitions of “algorithm” from Google, processed by wordclouds.com
What is an algorithm?

An **algorithm** for a function \( f : D \rightarrow R \) is a Turing machine \( M \), which given as input any \( d \in D \) on its tape, eventually halts with the correct answer \( f(d) \in R \) on its tape. Specifically, we can require that

\[
q_0 d \vdash^*_M q_f f(d), q_f \in F,
\]

for all \( d \in D \).
What is an algorithm?

Informally, an *algorithm* is any well-defined computational procedure that takes some value, or set of values, as *input* and produces some value, or set of values, as *output*. An algorithm is thus a sequence of computational steps that transform the input into the output.
Examples of famous algorithms

• Please take 30 seconds to think of one or two algorithms that you have heard of
Examples of famous algorithms

• Quicksort
• PageRank
• Fast Fourier transform
• Euclid’s algorithm (GCD)
• Dijkstra’s algorithm (shortest path in a graph)

Next: “nuts-and-bolts” algorithms versus “niche” algorithms
“Nuts-and-bolts” algorithms are used as building blocks in most computer programs

• Examples: sorting algorithms, hash tables, ...
“Niche” algorithms solve more specific problems

• A personal selection:
  • Decision trees
  • Error correcting codes
  • Compression
  • Digital signatures

Next: examine two specific algorithms to understand more details:
1. 2D parity error correcting code
2. Grade school multiplication
Without error correction, your upload to Facebook would look like this

PISA 2015 Results (Volume I), http://dx.doi.org/10.1787/19963777
One solution: 2D parity error correcting code

4 8 3 7 2 5 4 3 6 8 2 7 5 6 5 3 9 9 7 8 4 3 0 6

<table>
<thead>
<tr>
<th>4</th>
<th>8</th>
<th>3</th>
<th>7</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4</th>
<th>8</th>
<th>3</th>
<th>7</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>0</td>
<td>6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This “2D parity” algorithm has many connections to mathematics (and most other algorithms have similar mathematical relevance)

• Modular arithmetic: checksum is computed modulo 10

• Algebra: formula for corrected digit

• Proof: Is it guaranteed to correct any single error? Detect multiple errors?

• Probability and statistics: what is the chance of an undetected error?
The “grade school multiplication” algorithm provides another useful example

• If we double the number of digits, how much longer does it take?

• This is an example of complexity theory
Summary: what is an algorithm?

- Method for solving a problem using a computer program
- Usually based on mathematical techniques
Why should citizens know about algorithms?
Outline: life and learning in the age of algorithms

• Are we living in an age of algorithms?

• What is an algorithm?

• What algorithmic ideas should be in the school-age maths curriculum?
What algorithmic ideas should be in the school-age maths curriculum?

• Perhaps, none
  • i.e. algorithmic thinking is important, but not important enough to displace essential maths

• If we do want algorithms, a wide spectrum of approaches is possible
  • Two extremes on this spectrum:

  - **Unplugged**
    - Algorithms with no programming, no computers
    - Example: CSunplugged.org

  - **Integrated**
    - Algorithms fully integrated with programming and maths
    - Example: bootstrapworld.org
Teaching algorithmic thinking with no computers and no programming (CSunplugged.org)

Yes... this is QuickSort!

https://classic.csunplugged.org/sorting-algorithms/
A classic compression algorithm from CSunplugged.org

Yes... this is LZ compression, as used in ZIP files!
A classic compression algorithm from CSunplugged.org

Yes... this is LZ compression, as used in ZIP files!

https://classic.csunplugged.org/text-compression/
A classic compression algorithm from CSunplugged.org

Yes... this is LZ compression, as used in ZIP files!

https://classic.csunplugged.org/text-compression/
Example: the bootstrapworld.org Algebra course

- Maths + programming:
  - Pencil-and-paper workbooks
  - Write code in browser

- Algebra course content:
  - Cartesian coordinates
  - Functions, domain, range
  - Derive, discuss, and prove the Pythagorean theorem
  - Then *use* the Pythagorean theorem to detect collisions in a video game

---

```
(define (fact n)
  (cond
   [(< n 2) 1]
   [else (+ (fact (- n 1)) (fact (- n 2)))]))

(define (update-world w) (+ w 10))

(define (draw-world w)
  (begin
    (fact DIFFICULTY)
    (rotate (modulo w 360) img)))
```
Other examples of integrating mathematics with programming

• Draw n-gon in Scratch

• Integration via trapezoidal approximation

https://scratch.mit.edu/projects/12600956
Both ends of the spectrum have advantages

**Unplugged** (less) ... programming ... (more) **Integrated**

**Advantages:**
- No setup time
- No software problems
- Low barrier for instructors

**Advantages:**
- STEM equity (all students exposed to code+maths)
- Potentially superior learning outcomes due to active learning
Summary: life and learning in the age of algorithms

• Are we living in an age of algorithms?
  • Yes, but don’t confuse algorithms with technology

• What is an algorithm?

• What algorithmic ideas should be in the school-age maths curriculum?
  • 3 possible approaches: None, Unplugged, Integrated