Algorithms

Informally, an algorithm is ...

A well-defined computational procedure that takes some value, or set of values, as \textit{input} and produces some value, or set of values, as \textit{output}.

A sequence of computational steps that transform the \textit{input} into \textit{output}.
Algorithms

Empirically, an algorithm is...

A tool for solving a well-specified computational problem.

Problem specification includes what the input is, what the desired output should be.

Algorithm describes a specific computational procedure for achieving the desired output for a given input.

The Sorting Problem:

Input: A sequence of $n$ numbers $[a_1, a_2, \ldots, a_n]$.
Output: A permutation or reordering $[a'_1, a'_2, \ldots, a'_n]$ of the input sequence such that $a'_1 \leq a'_2 \leq \ldots \leq a'_n$.

An instance of the Sorting Problem:

Input: A sequence of 6 number $[31, 41, 59, 26, 41, 58]$.

Expected output for given instance:

Algorithms

Some definitions ...

An algorithm is said to be **correct** if, for every input instance, it **halts** with the correct output.

A **correct** algorithm **solves** the given **computational problem**.

Focus will be on **correct** algorithms; incorrect algorithms can sometimes be useful.

Algorithm specification may be in English, as a computer program, even as a hardware design.

---

Gallery of Problems

Algorithms are needed (most of which are novel) to solve the many problems listed here ...

The Human Genome Project seeks to identify all the 100,000 genes in human DNA, determining the sequences of the 3 billion chemical base pairs comprising human DNA, storing this information in databases, and developing tools for data analysis.

The huge network that is the Internet and the huge amount of data that courses through it require algorithms to efficiently manage and manipulate this data.
Gallery of Problems

E-commerce enables goods and services to be negotiated and exchanged electronically. Crucial is the maintenance of privacy and security for all transactions.

Traditional manufacturing and commerce require allocation of scarce resources in the most beneficial way. Linear programming algorithms are used extensively in commercial optimization problems.

Some algorithms

- Shortest path algorithm
  - Given a weighted graph and two distinguished vertices -- the source and the destination -- compute the most efficient way to get from one to the other

- Matrix multiplication algorithm
  - Given a sequence of conformable matrices, compute the most efficient way of forming the product of the matrix sequence
Some algorithms

- Convex hull algorithm
  - Given a set of points on the plane, compute the smallest convex body that contains the points

- String matching algorithm
  - Given a sequence of characters, compute where (if at all) a second sequence of characters occurs in the first

Hard problems

- Usual measure of efficiency is speed
  - How long does an algorithm take to produce its result?
  - Define formally measures of efficiency

- Problems exist that, in all probability, will take a long time to solve
  - Exponential complexity
  - NP-complete problems

- Problems exist that are unsolvable
Hard problems

- NP-complete problems are interesting in and of themselves
  - Some of them arise in real applications
  - Some of them look very similar to problems for which efficient solutions do exist
  - Knowing the difference is crucial
- It is not known whether NP-complete problems really are as hard as they seem, or, perhaps, the machinery for solving them efficiently has not been developed just yet

---

Hard problems

- \( P \neq NP \) conjecture
  - Fundamental open problem in the theory of computational complexity
  - Open now for 30+ years
Algorithms as a technology

- Even if computers were infinitely fast and memory was plentiful and free
  - Study of algorithms still important – still need to establish algorithm correctness
  - Since time and space resources are infinite, any correct algorithm would do
- Real-world computers are fast but not infinitely so
- Memory is cheap but not unlimited

Efficiency

- Time and space efficiency are the goal
- Algorithms often differ dramatically in their efficiency
  - Example: Two sorting algorithms
    - Insertion-Sort – time efficiency is $c_1n^2$
    - Merge-Sort – time efficiency is $c_1n\log n$
  - For which problem instances would one algorithm be preferable to the other?
Efficiency

- Answer depends on several factors:
  - Speed of machine performing the computation
    - Internal clock speed
    - Shared environment
    - I/O needed by algorithm
  - Quality of implementation (coding)
    - Compiler optimization
    - Implementation details (e.g., data structures)
  - Size of problem instance
    - Most stable parameter – used as independent variable

- **INSERTION-SORT**
  - Implemented by an ace programmer and run on a machine A that performs $10^9$ instructions per second such that time efficiency is given by:
    \[ t_A(n) = 2n^2 \text{ instructions (i.e., } c_1=2) \]

- **MERGE-SORT**
  - Implemented by a novice programmer and run on a machine B that performs $10^7$ instructions per second such that time efficiency is given by:
    \[ t_B(n) = 50n\log n \text{ instructions (i.e., } c_1=50) \]
Efficiency

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Machine A Insertion-Sort</th>
<th>Machine B Merge-Sort</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>$2n^2/10^9$</td>
<td>$50n\log n/10^7$</td>
</tr>
<tr>
<td>10,000</td>
<td>0.20</td>
<td>0.66</td>
</tr>
<tr>
<td>50,000</td>
<td>5.00</td>
<td>3.90</td>
</tr>
<tr>
<td>100,000</td>
<td>20.00</td>
<td>8.30</td>
</tr>
<tr>
<td>500,000</td>
<td>500.00</td>
<td>47.33</td>
</tr>
<tr>
<td>1,000,000</td>
<td>2,000.00</td>
<td>99.66</td>
</tr>
<tr>
<td>5,000,000</td>
<td>50,000.00</td>
<td>556.34</td>
</tr>
<tr>
<td>10,000,000</td>
<td>200,000.00</td>
<td>1,162.67</td>
</tr>
<tr>
<td>50,000,000</td>
<td>5,000,000.00</td>
<td>6,393.86</td>
</tr>
</tbody>
</table>

Graphical comparison

- Time Efficiency Comparison
Algorithms vis-à-vis other technologies

- Are algorithms really that important in the face of dramatic advances in other technologies?
  - Hardware: superfast clock speeds, parallelism, pipelining
  - Graphical User Interfaces (GUI)
  - Object Oriented Systems
  - LANs and WANs

- YES! Algorithms are at the core of most technologies used in contemporary computation